UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

ORACLE AMERICA, INC. Plaintiff, v. GOOGLE, INC. Defendant.	Case No. 3:10-cv-03561-WHA

OPENING EXPERT REPORT OF ANDREW HALL

TABLE OF CONTENTS

I. INTRODUCTION

A. Case Background

1. I have been retained as an independent expert on behalf of Google in this litigation. My billing rate for this litigation is $500 per hour, the same 2015 hourly rate charged to Hall Law clients. My compensation does not depend in any way on the opinions I provide, nor do I have any direct financial interest in this case.[1]

2. Google has engaged me to apply my expertise in the field of open-source licensing to explain key related concepts and their relevance to the issues in the current case. To that end, I provide some background regarding open-source software licensing, including common open-source licenses, business reasons for publishing (or "releasing") software under an open-source license ("open-sourcing" software), and a comparison of particular open-source software licenses to aid the jury in the evaluations of various issues in the case, including fair use and damages. I also explain the Apache License version 2 ("Apache-2.0 license"), as well as the GNU General Public License version 2 ("GPL-2.0 license") and its related exceptions, including the Classpath Exception ("GPL-2.0-CE license"). Finally, I have also been asked to evaluate Google's Android project including OpenJDK software and opine on technical implementation issues related to software "linking."

3. At this time, I have not created any demonstrative exhibits to be used as a summary of or support for my opinions, setting aside the graphics in this report. I reserve the right to create any additional summaries, tutorials, demonstrations, charts, drawings, tables, and/or animations that may be appropriate to supplement and demonstrate my opinions if I am asked to testify at trial.

[1] I do, however, own broad index and mutual funds that likely include shares of Oracle and/or Google stock.

4. I understand that discovery is ongoing in this case. I therefore reserve the right to supplement my opinions after I have had the opportunity to review deposition testimony or in light of additional documents that may be brought to my attention. I further understand that the Court has adopted an expert-discovery schedule that allows experts to file rebuttal and/or reply reports, depending on the burdens of proof. Notwithstanding that schedule, if Oracle or its experts change their opinions (either explicitly or implicitly) in such a manner as to affect my conclusions, I may supplement my opinions with any necessary rebuttal reports.

B. Professional Qualifications

5. I am a founding Partner of Hall Law, a boutique law firm established in January 2015 and focused on legal and strategic issues surrounding the development, release, commercialization, and governance of both commercial and open-source hardware and software as well as issues relating to intellectual property ("IP") governance.

6. Examples of open-source counseling and services that Hall Law commonly provides to clients include: (a) preparing software development guides, compliance guides, and other internal client documentation designed to assist clients' employees and contractors in adopting "best practices" with respect to the use and commercialization of open-source software; (b) conducting audits and reviewing third-party audits (conducted by audit and compliance services providers such as Palamida, Black Duck, Protecode, OpenLogic, and NexB) to assess the patent, copyleft (defined below), and compliance implications of proposed or actual client use of open-source software; (c) advising clients on the incorporation of open-source software into commercial and closed-source products and services and other commercial dependencies upon open-source software; (d) developing and consulting with clients to develop internal governance policies and practices for procuring open-source software, incorporating open-source software into clients products and services, releasing proprietary client software

under an open-source license, and complying with any obligations resulting from such use or release of open-source software; (e) consulting with and negotiating on behalf of clients engaged in an investment event, such as a merger, acquisition, funding, or initial public offering, in diligence matters and negotiations relating to open-source software; (f) counseling and negotiating on behalf of clients engaged in open-source compliance disputes or that have otherwise received an infringement claim or invitation to license stemming from the clients' use of open-source software; (g) preparing open-source training materials and providing open-source training sessions for clients' employees and contractors including, for example, hardware and software engineers, legal staff, executives, and management; and (h) advising clients on the strategic release of proprietary client software under the terms of open-source licenses such as the Apache-2.0 license, GPL-2.0 license, or GPL-2.0-CE licenses. I provided counseling on the same topics while with Fenwick & West from 2010-2015.

7. I received a Juris Doctor degree from University of Chicago Law School in 2006. I practiced law with Knobbe Martens Olson & Bear, focusing primarily on IP litigation, from 2006 to 2010. I practiced law with Fenwick & West, focusing on technical transactions and the commercialization of closed-source and open-source software in particular, from 2010 to 2015.

8. I am also a trained computer scientist and engineer. I received a Bachelor of Science Electrical Engineering in 2001 from The Ohio State University in Electrical and Computer Engineering. From 2001 to 2003 I worked as a technical employee of National Instruments, Dell, and Raven Industries My duties at each company included one or more of: (a) product development, testing, and repair; (b) providing technical support to customers (such as researchers and engineers), technical service providers (such as contract IT technicians), and customers' IT staffs relating to company technical products and services and managing

customer relationships with respect to the same; and (c) leading courses and other training sessions relating to company technical products and services (such as measurement and automation hardware and software and commercial-grade GPS systems).

9. I have published various articles discussing open-source software, including the following:

a) *Free and Open-Source Software Diligence in Mergers, Acquisitions, Public Offerings, and Other Investments*, CEB California Business Law Practitioner, January 2014.

b) *Free and Open-Source Software Diligence in Mergers, Acquisitions, and Investments*, Lexology, December 2012.

c) *Free and Open-Source Software Diligence in Mergers, Acquisitions, and Investments*, Fenwick & West IP Bulletin, August 2012

10. In addition, my article *Open-Source Licensing and Business Models: Making money by giving it away* has been slated for inclusion in the Santa Clara Law School's High Tech Law Journal.

11. I have led or participated in various presentations and lectures relating to legal issues surrounding open-source software, including the following:

a) *Intersections of Patents and Open-Source Software*, Palamida Webinar, June 2015;

b) *Open-Source License Enforcement*, Palamida User Group, September 2015;

c) *Open-Source Licensing and Business Models: Making money by giving it away*, keynote speaker, Santa Clara High Tech Law Journal Open-Source Symposium, January 2015;

d) *Patent Pledges and other Open-Source Patent Grants*, Asian Legal Network, December 2014;

e) *Free and Open-Source Licensing and Business Models*, All Things Open, October 2014.

f) *An Introduction to Free and Open Source Software Licensing*, Open Business Conference 2014, May 2014;

g) *Profiting with Open Source: An Introduction to Open Source Licensing and Business Models*, Great Wide Open 2014, April 2014;

h) *An Introduction to Free and Open Source Software Licensing*, Linux Foundation Collaboration Summit 2014, March 2014;

i) *The Importance of Open Source Due Diligence in M&A*, Black Duck Webinar, January 2014;

j) *An Introduction to Free and Open Source Software Licensing*, Palamida Webinar, December 2013;

k) *The Changing Compliance Landscape: Advanced Open Source Issues for Large Enterprises*, Open Business Conference 2013, April 2013;

l) *Free and Open-Source Software Diligence in Mergers and Acquisitions*, Linux Foundation Collaboration Summit 2013, April 2013;

m) *Open-Source Best Practices*, Palamida User Group, March 2013

n) *Open-Source Software Diligence in Mergers, Acquisitions and Investments*, Palamida webinar, February 2013; and

o) *The Changing Compliance Landscape: How Innovations Like Maven, Ruby Gems, GitHub, and IaaS/PaaS Present New Challenges for Open Source Licensing and Compliance*, Open Business Conference 2012, May 2012.

12. My curriculum vitae is attached as Exhibit A.

C. Materials Considered

13. In forming my opinions and preparing this report, I have considered the materials cited and listed in this report, as well as the documents listed in Appendix B.

14. In addition, I spoke with Anwar Ghuloum, Engineering Manager of the Android Runtime & Tools team at Google, and Google Senior Software Engineer Brian Carlstrom regarding the development of a version of Android based off of OpenJDK.

15. I also have specialized knowledge of the matters set forth herein, and if called as a witness I could and would testify competently thereto.

16. My research and analysis of the materials, documents, allegations, and other facts in this case are ongoing, and if additional information becomes available through discovery, I reserve the right to supplement my opinions.

II. SUMMARY OF OPINIONS

17. Among other things, "free" or "open source" software licenses generally make source code publicly available, allow recipients to modify and redistribute the software, and prohibit discriminating against particular uses of the license software (*e.g.*, commercial use).

18. An open-source software license can be classified by its copyleft effect as either "permissive," "weak-copyleft," or "strong-copyleft." Examples of permissive licenses (also known as "attribution licenses") include the BSD license, the MIT license, and the Apache-2.0 License. Permissive licenses impose few obligations (usually limited to the delivery of a copy of the license) on the use and distribution of the licensed software.

19. Copyleft licenses (sometimes referred to as "reciprocal" or viral" licenses) impose obligations and restrictions (together "requirements") on the redistribution of the licensed software that include not only the type of notice obligations imposed by permissive licenses, but also a set of "copyleft" requirements. Copyleft requirements have two essential elements: (1) any licensee that redistributes the licensed software in binary code (or another *non*-source) form must also make the licensed software available to recipients in source code form; and (2) the corresponding source code must be distributed only under the terms of the original copyleft license.

20. There are variations among copyleft licenses that are important to my analysis. The most common copyleft license, the GNU General Public License ("GPL"), is sometimes referred to as a "strong" copyleft license because it requires any *derivative work* under copyright law (which the version 2 of the GPL ("GPL-2.0") interprets to include any programs that contain

the licensed software, in whole or in part) to be made available in source code form under the same license.

21. Another class of licenses, so-called "weak" copyleft licenses, impose similar copyleft requirements on redistributors of the licensed software but explicitly exclude certain combinations of software from copyleft requirements, meaning the combined software can be distributed in binary code (or any other) form under the terms of the licensee's choice (including closed-source and other typical end-user terms) without obligation to make the source code corresponding to the combined software available in source code form under the weak-copyleft license. Thus, weak-copyleft licenses provide developers with approved ways to mix and match the weak-copyleft-licensed software with software under *non*-copyleft, closed-source, and other limited terms. Weak-copyleft licenses include the Library or Lesser General Public License ("LGPL") as well as the GPL-2.0 license with the Classpath Exception ("GLP-2.0-CE"), which is designed to allow linking to the licensed software without subjecting the linking software to the same open-source license terms.

22. Various commercial products include source code provided under all three categories of licenses. In fact, software and technology products frequently contain at least some open-source software. Examples include end-user electronic devices sold by Apple, Microsoft, Samsung, Sony, and Panasonic, and software products made available by Microsoft, Adobe, Apple, and others. The fact that such software is provided under an open-source license neither prohibits commercial use nor makes it impractical for such use.

23. The optimal open-source license category and license for an open-source software package employed as part of an open-source business model depends on the role that software package plays within the business model and the manner in which others will use the software

package. Weak-copyleft licenses are often an ideal selection for open-source software libraries. Libraries are collections of software packaged in a manner that enables reuse of the software collection as part or in support of another software program. A weak-copyleft license obligates downstream distributors to make distributed copies of the licensed library (including any modifications introduced by the distributor) available under the same weak-copyleft license. That requirement enables others in the software community (including the original weak-copyleft licensor) to use the distributor's modifications in combination with the original weak-copyleft licensed software library under the same licensing terms that apply to the original unmodified software library. In this way, the entire open-source community benefits from any modifications made by an individual library distributor. However, weak-copyleft licenses also allow developers to combine a licensed software library with other open-source or closed-source software programs without concern that their combined software programs will need to be made available in source code form under the terms of the weak-copyleft license covering the library. In this way, weak-copyleft licenses allow developers flexibility to use the software library as part of larger software programs and collections regardless of whether the programs or collections are provided under the same or different licensing terms.

24. Depending on the applicable license terms, many open-source software packages can be distributed as part of a single package or product despite their being distributed under different open-source licenses. In other words, open-source software can often be mixed and matched within different products, as evidenced by a variety of real-world examples.

25. Google has recently revised the Android platform to use software included in OpenJDK, Oracle's open-source release of the Java platform that implements the standard language libraries necessary to run applications written in the Java programming language for

the Android platform. (Google previously implemented these API packages based on source code from the Apache Harmony project.) As to the accused 37 Java API packages from OpenJDK as opposed to Apache Harmony, Google has the right to use, modify, and redistribute those packages, as included in Android, under the GPL-2.0-CE license (which is the license Oracle applies to OpenJDK). However, this change does not conflict with the open-source licenses applicable to other Android components; for example, other API packages can remain under the Apache-2.0 license.

26. The GPL-2.0-CE license under which Oracle (in part via its predecessor-in-interest to Java, Sun Microsystems) open-sourced the OpenJDK implementation of Java allows loading, linking, and executing the OpenJDK-based Java API packages without subjecting the linking software to the terms of the GPL-2.0 license. Specifically, under the Classpath Exception, Google, its OEMs, and Android application developers can all link to Android's OpenJDK-based API packages without having to place their own linking software under the terms of the GPL-2.0-CE or GPL-2.0 license.

27. Google could have licensed and modified the 37 API packages from OpenJDK (as opposed to Apache Harmony) when OpenJDK was first released under the GPL-2.0-CE license as of 2007.

III. OPEN-SOURCE LICENSING GENERALLY

A. Software Packaging

28. The term software "package" is often used to refer to a series of files, classes, and interfaces (including, for example, source code, bytecode, or binary code and related documentation) collected into one or more distributable files, libraries, plug-ins, or software "archives." The term is sometimes used broadly to refer to an independent software application (such as a word processor), an operating system (such as Linux), or even an entire software

"stack" (such as the LAMP stack[2]) which enables separately developed software to execute on devices on which the stack has been installed. The term "package" is sometimes used more narrowly to refer to a smaller collection of files providing related functionality or even a single class file. It is not uncommon for a software package to either include or require one or more other software packages ("sub-packages") that the software package uses during execution. These can include, for example, a runtime platform or engine on which the package runs and software libraries providing the package with dedicated functionality.

29. It is not uncommon for software packages to be distributed in combination with other software packages or sub-packages that are licensed to the distributor or by the distributor to recipients under different licensing terns. Software distributed in combination can include collections of open-source software packages distributed under different licenses (such as the LAMP stack), collections of closed-source software provided under different licenses or terms (such as an application with supporting database software licensed under a limited end-user license), or a combination of closed-source and open-source packages (such as software products that incorporate or depend upon open-source packages). Examples of closed-source products and services including open-source packages are provided in the following table.

Manufacturers	**Product**	**Open Source Licensing**
Apple	Smartphones, tablets, and portable media players	Additional details regarding the use of open-source software in Apple products are available at https://opensource.apple.com/

[2] A large collection of software that includes open-source packages Linux, Apache Web Server, MySQL, and PHP/Python/Perl and is used as the backbone for many websites.

Manufacturers	Product	Open Source Licensing
Microsoft	Smartphones, tablets, operating systems, consumer software application (e.g., MS Office)	Additional details regarding the use of open-source software in Microsoft products are available at: http://thirdpartysource.microsoft.com/; https://www.microsoft.com/hardware/en-us/support/oss-license
Samsung	Tablets, TVs, Blu-ray recorders, set-top boxes, MP3 players, digital cameras, camcorders, printers, monitors, mobile digital x-ray machines.	Additional details regarding the use of open-source software in Samsung products are available at: https://opensource.samsung.com/
LG	Smartphones, tablets, air conditioners, washing machines, refrigerators, microwaves, car infotainment systems, media players, security systems, and network storage	Additional details regarding the use of open-source software in LG products are available at: http://opensource.lge.com/.
HTC	Smartphones and tables	Additional details regarding the use of open-source software in HTC products are available at: http://www.htcdev.com/devcenter/downloads
Sony	Smartphones, tablets, set-top boxes, TVs, navigation systems, video game consoles, media players, home entertainment	Additional details regarding the use of open-source in Sony products are available at: http://developer.sonymobile.com/downloads/opensource/ https://products.sel.sony.com/opensource/ http://oss.sony.net/Products/Linux/common/search.html
Blackberry	Smartphones and tablets	Additional details regarding the use of open-source in Blackberry products are available at: http://blackberry.github.io/ http://us.blackberry.com/software/smartphones/blackberry-6-os/browser-open-source-components.html http://devblog.blackberry.com/category/open-source-2/

Manufacturers	**Product**	**Open Source Licensing**
Amazon.com	Tablets, PCs, e-readers, streaming video, and streaming media players	Additional details regarding the use of open-source in Amazon products are available at: http://www.amazon.com/gp/help/customer/display.html?nodeId=200203720 http://www.amazon.com/gp/help/customer/display.html?nodeId=201422780 http://www.amazon.com/gp/help/customer/display.html?nodeId=201497520 http://www.amazon.com/gp/help/customer/display.html?nodeId=201452680
Garmin	GPS navigation systems and mobile apps	Additional details regarding the use of open-source in Garmin products are available at: http://developer.garmin.com/open-source/overview/
TomTom	GPS navigation systems and supporting apps	Additional details regarding the use of open-source in TomTom products are available at: https://www.tomtom.com/en_gb/opensource/
Panasonic	TVs, blu-ray disc players and recorders, home theater components, hard disk recorders, portable AV devices, digital cameras and video cameras, tablets, set-top boxes, and payment terminals.	Additional details regarding the use of open-source in Panasonic products are available at: http://panasonic.net/avc/oss/index.html
Netgear	Network routers, modems, adapters, hotspots, and storage, home security systems.	Additional details regarding the use of open-source in Netgear products are available at: http://kb.netgear.com/app/answers/detail/a_id/2649
Toyota/Lexus	Automobiles	Additional details regarding the use of open-source in Toyota/Lexus products are available at: http://www.globaldenso.com/en/opensource/ivi/toyota/

Manufacturers	**Product**	**Open Source Licensing**
Facebook	iOS and Android apps	For additional details regarding the use of open-source in Facebook's iOS app select "More" > "Terms & Policies" > "More Resources" > "Third Party Notices" from within an installed iOS Facebook app.
Twitter	iOS and Android apps	For additional details regarding the use of open-source in Twitter's iOS app select "Me" > [gear icon] > "Settings" > "About" > "Legal" within an installed iOS Twitter app.
LinkedIn	iOS and Android apps	Additional details regarding the use of open-source in LinkedIn's mobile apps are available at: https://www.linkedin.com/legal/mobile/open-source-mobile-apps
Dropbox	Desktop, iOS, Android apps	Additional details regarding the use of open-source in Dropbox's mobile apps are available at: https://www.dropbox.com/opensource/dropbox/ios https://www.dropbox.com/opensource/dropbox/android
Adobe	Photoshop, Acrobat, and other desktop applications	Additional details regarding the use of open-source in Adobe products are available at: https://www.adobe.com/products/eula/third_party/.

B. Free and Open Source Licensing

30. The term "free software" was popularized by the Free Software Foundation (the "FSF")[3] and is used to refer to licenses that provide the four "freedoms" identified in the FSF's "Free Software Definition," which are:

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1).[4]

[3] The Free Software Foundation is a nonprofit organization "with a worldwide mission to promote computer user freedom and to defend the rights of all free software used." FSF homepage: https://www.fsf.org/about/
[4] The FSF concludes: "[a]ccess to the source code is a precondition for [satisfying freedom 1]."

- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3).[5]

The Free Software Definition is published at: https://www.gnu.org/philosophy/free-sw.en.html.

31. The term "open-source software" was popularized by the Open Source Initiative (the "OSI")[6] and is used to refer to licenses that meet the ten criteria identified in the OSI's "Open Source Definition," which are:

- **Free Redistribution.** The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.
- **Source Code.** The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.
- **Derived Works.** The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
- **Integrity of the author's source code**. The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

[5] The FSF concludes: "by [permitting distribution of modified versions] you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this." *Id.*
[6] The OSI is a California public benefit corporation with 501(c)(3) tax-exempt status. Founded in 1998, the OSI is actively involved in Open Source community-building, education, and public advocacy to promote awareness and the importance of non-proprietary software. OSI homepage: https://opensource.org/

- **No discrimination against persons or groups.** The license must not discriminate against any person or group of persons.

- **No discrimination against fields of endeavor.** The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

- **Distribution of license.** The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

- **License must not be specific to a product**. The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

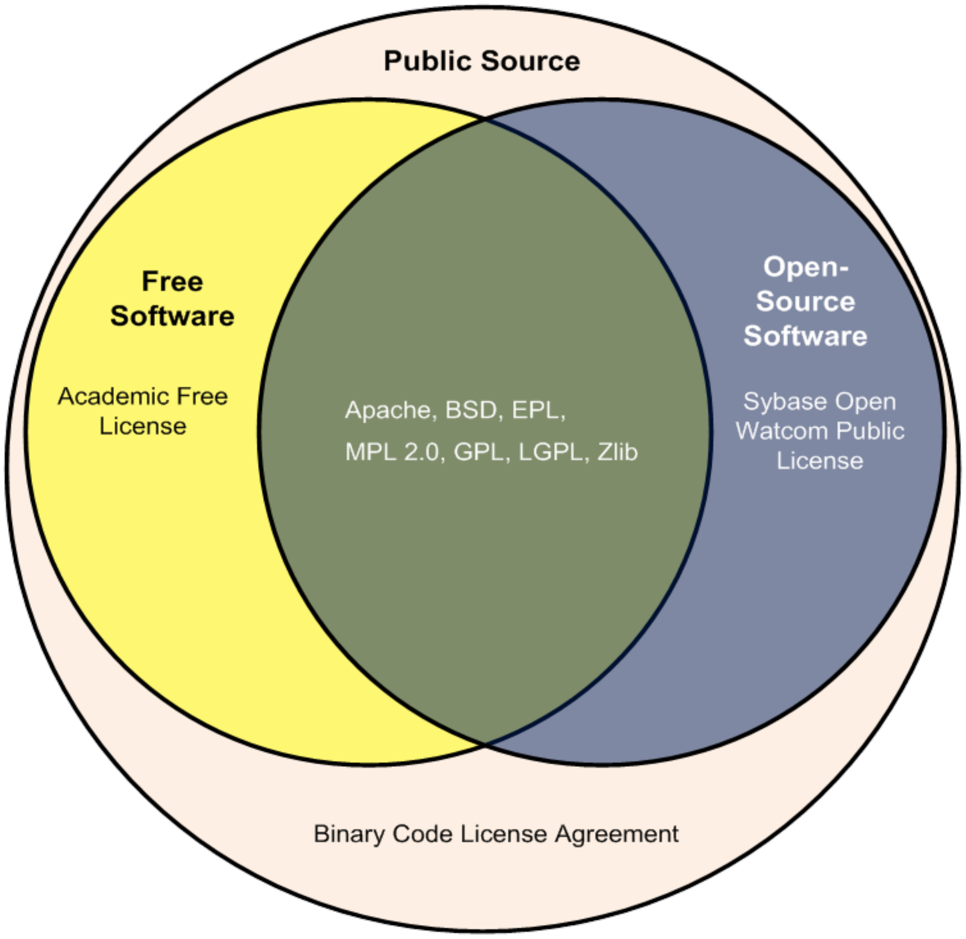
- **License must not restrict other software.** The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

- **License must be technology-neutral.** No provision of the license may be predicated on any individual technology or style of interface.

The Open Source Definition is published at: https://opensource.org/osd. An Open Source Definition annotated by the OSI is published at: https://opensource.org/osd-annotated.

32. Although the FSF and OSI definitions are not coextensive, they do place some of the same requirements on qualifying licenses including: (a) the availability of the licensed software's source code; (b) the recipient's rights to modify and redistribute the licensed software; and (c) a prohibition on discriminating against particular uses of the licensed software. Furthermore, despite the imperfect correlation, the terms "free software" and "open-source software" are colloquially often used interchangeably or referred to collectively as "open source," "FOSS," or "FLOSS".

33. In contrast to "open source" software, "public source" software refers to the broader class of software that is made available (a) to the public, (b) in source code form, and (c) under the terms of a standard, royalty-free license. The diagram below depicts the overlap between free software, open-source software, and public-source software licenses and provides examples of licenses falling into each category.


Public Source
Free Software
Academic Free License
Apache, BSD, EPL, MPL 2.0, GPL, LGPL, Zlib
Open-Source Software
Sybase Open Watcom Public License
Binary Code License Agreement

C. The Copyleft Effect of Certain Licenses

34. "Copyleft" was coined as a play on the term "copyright" and refers generally to the philosophy espoused by the FSF to limit the use of copyrights and patents to restrict the free modification, copying, and distribution of software. This copyleft philosophy is embodied by the FSF's GNU General Public License ("GPL"), which requires distributors of GPL-licensed

software ("GPL software") to make the source code for both the GPL software and any work derived from the GPL software available for royalty-free use, copying, and further distribution under the terms of the GPL. Open-source licenses having this "copyleft effect" are sometimes also described as hereditary, reciprocal, or viral licenses.

D. Categorizing and Describing Open-Source Licenses

35. The potential copyleft effect of an open-source is crucial to distributors of closed-source software who are concerned about losing rights to their software or other commercially valuable IP. The copyleft effect can also deter distributors of closed-source products from deciding to include an open-source software package in their products.

36. An open-source license is often categorized as either "strong-copyleft," "weak-copyleft," or "permissive" based on the existence and scope of its copyleft effect. The requirements of most open-source licenses are triggered by distribution of the licensed open-source software.

1. Strong Copyleft

37. "Strong-copyleft" open-source licenses require that both the strong-copyleft software and any software that is a derivative work of (or "based on") the copyleft software be made available in source code form under the terms of the same strong-copyleft license. Most strong-copyleft licenses have a copyleft effect only on ***distributed*** derivative works of the copyleft software. The most commonly used strong-copyleft license is the GPL-2.0 license.

2. Weak Copyleft

38. "Weak-copyleft" open-source licenses (sometimes also referred to as "file-level" copyleft licenses) are intended to have a narrower or at least more specific copyleft effect. Most weak-copyleft licenses require that distributed copies of the weak-copyleft software, including

any modifications made by the distributor, be provided or made available to recipients in source code form under the terms of the same weak-copyleft license. However, when combined as permitted under the weak-copyleft license, the license does not extend these copyleft effects to the entire program with which the weak-copyleft software is combined or distributed. What constitutes a modification to the open-source software is often specified in detail within the applicable weak-copyleft license.

39. Weak-copyleft licenses permit at least dynamic (*aka* runtime) linking between weak-copyleft and closed-source software without imposing a copyleft effect on the linking closed-source software. Some also permit static (*aka* compile-time) linking without copyleft effect on the linking software. If closed-source software is combined with weak-copyleft software in a manner not authorized by the license, the weak-copyleft license can have broad copyleft effects similar to those of a strong-copyleft license.

40. Commonly used weak-copyleft licenses include the GPL-2.0-CE and LGPL licenses, Mozilla Public License (the "MPL"), Common Public License (the "CPL"), and Common Development and Distribution License (the "CDDL").

3. Permissive

41. "Permissive" open-source licenses do not have a copyleft effect. However, use of software under a permissive licenses often does impose at least some obligations or restrictions. Like strong-copyleft and weak-copyleft licenses, permissive open-source licenses often require distributors to provide recipients of the weak-copyleft software with attribution, copyright, and disclaimer notices, or a copy of the applicable open-source license. Frequently used permissive open-source licenses include the Apache, BSD, MIT, and zlib licenses.

42. The following table summarizes the different types of public-source licenses:

License Type:	Permissive	Weak Copyleft	Strong Copyleft

Examples:	Apache-2.0;
	MIT;
	BSD
		CDDL;
		EPL;
		LGPL;
		MPL;
		GPL-2.0-CE
			GPL;
			Sleepycat;
			OSL

Copyleft effect:	No copyleft effect under any circumstances.	Copyleft effect generally limited to ***modifications*** to distributed copies of the open-source package.	Copyleft effect extends to "***derivative works***" of distributed copies of the open-source package.
Notice obligation:	Permitted distributed use often requires providing recipients with certain notice relating to the package such as a copy of the applicable open-source license.	Permitted distributed use usually requires providing recipients with certain notices relating to the package such as a copy of the applicable open-source license or acknowledgment of the package's use.	
Prohibition on commercial use	No.		

43. Public-source licenses forbidding the commercial use of the licensed software are prohibitive and do not qualify as either a "free-software" or "open-source" license. Such limitations and prohibitions conflict with (a) the first of the four required freedoms in the Free Software Definition, "the freedom to run the program as you wish for any purpose,"[7] and (b) the sixth enumerated criterion of the Open Source Definition: "No discrimination against fields of endeavor."[8]

[7] FSF.org, *What is free software?* Published at: http://www.gnu.org/philosophy/free-sw.en.html

[8] OpenSource.org, The Open Source Definition (Annotated), https://opensource.org/osd-annotated ("**6. No Discrimination against Fields of Endeavor.** The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research. ***Rationale:*** *The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.*")

E. The Apache 2.0 License

44. The Apache-2.0 license (published by the Apache Software Foundation in 2004) is a ***permissive*** license, which means the Apache-2.0 license does not have a copyleft effect on modifications to, derivative works of, or combinations with Apache-2.0 software under any use scenario. Thus, regardless of how Apache-2.0-licensed software is used or combined with software licensed under different terms, the Apache-2.0 license will not obligate users or distributors to (a) provide or offer to provide recipients with corresponding source code for the Apache-2.0 licensed software; or (b) license modifications to or derivative works of the Apache-2.0 licensed software under the Apache-2.0 license or any other open-source license. The Apache-2.0 license has been certified as "open source" by the OSI and as "free software" by the FSF.

45. The table below identifies a few of the popular open-source projects licensed under the Apache-2.0 license as well as the companies sponsoring and contributing to those projects.

Project	**Companies sponsoring and contributing to the project**
Hadoop https://hadoop.apache.org/	Microsoft, Twitter, Facebook, LinkedIn, Intel, IBM, eBay, Huawei, Yahoo!, VMware, Horton works, Cloudera, InMobi
HTTP server https://httpd.apache.org/	Facebook, Microsoft, Google, HP, Comcast, Bloomberg, IBM, Huawei, RedHat, Twitter, Samsung, Syamntec
OpenStack https://www.openstack.org/	AT&T, HP, IBM, Intel, RedHat, Cisco, Dell, Fujitsu, Hitachi, Huawei, NetApp, Symantec, Yahoo!, Accenture, Comcast, Oracle, SAP, Paypal, VMware.

46. The table below identifies commercial products that incorporate software licensed under the Apache-2.0 license:

Product	**Apache-2.0 projects**
Oracle Server X5-2 https://docs.oracle.com/cd/E41059_01/html/E48322/gpvur.html	Apache Commons Code, Apache Commons JXpath, Apache Commons Logging, Apache Jackarta Commons HttpClient,
Oracle JD Edwards Enterprise One https://docs.oracle.com/cd/E24705_01/doc.91/e58569/third_party.htm#EOTLI139	Commons BeanUtils, Commons Codec, Commons Digester, Commons HttpClient, Commons Lang, Commons Logging, Commons Net, Commons Validator, Jakarta Commons Daemon, Jakarta Commons Modeler, SOAP, Xerces, Xerces Java, Xalan
Oracle Health Sciences Empirica Signal http://docs.oracle.com/cd/E57976_01/doc.7334/e57744/e57744_01.htm	Apache Commons DBCP, Apache Commons FileUpload, Apache Commons Pool, Apache Jakarta Commons FileUpload, Apache Jakarta Commons Logging, Apache Log4J, Apache POI, Apache Tomcat, Apache Xerces
Oracle DIVAdirector https://docs.oracle.com/cd/E64802_01/DIVLG/licensing_info.htm#A1012444	log4net, Topshelf, PostgreSQL
Oracle Hierarchical Storage Manager and StorageTek QFS Software http://docs.oracle.com/cd/E60433_01/en/E64635/html/licensing_info.htm	Apache Batik, Apache Java Server Faces (JSF), Apache Tomcat
Oracle Ethernet Switch ML2 Mechanism Driver https://docs.oracle.com/cd/E60179_01/html/E64529/makehtml-id-3.html	expect4j, jakarta-oro
SalesforceIQ iOS and Android apps and Chrome Extension https://help.salesforce.com/servlet/servlet.FileDownload?file=0153000000382fbAAA	User Voice, gtm-oauth2, MKNumberBadgeView, RE2, JSONKit, Volley, RoundedImageView, Jackson, ProgressDialogTask, Snackbar, thirft.js, Unicode.js
Apple Mac OS X, Server (operating systems), and Xcode Tools (developer tools) http://www.apple.com/opensource/	Apache, apache_mod_bw, apache_mod_encoding2, apache_mod_jk, mod_perl, php, apache_mod_python, apache_mod_scgi_pubsub, apache_mod_xsendile

Hitachi Device Manager https://www.hds.com/assets/pdf/hitachi-device-manager-open-source-software-packages.pdf	Axis, Commons-beanutils, Commons-codec, Commons-collections, Commons-dbcp, Commons-digester, Commons-discovery, Commons-fileupload, Commons-httpclient, Commons-io, Commons-lang, Commons-logging-api, Commons-pool, Commons-validator, Apache FOP, cglib, ehcache, Fabrications, Jakarta-oro.jar, JSONIC, LOG4J, Quarts, Servlet API, Spring, Struts.
Cisco Security Manager http://www.cisco.com/c/en/us/td/docs/security/security_management/cisco_security_manager/security_manager/4-0/installation/guide/instl_wrapper/open_source.html#wp1015625	Beanutils, CLI, Collections, DbUtils, Digester, Lang, Logging
Hewlett Packard Arcsight http://www8.hp.com/h20195/v2/GetPDF.aspx/4AA6-0822ENW.pdf	Logging, Masukomi, Qdox, Xerces Java Parser, Tomcat, Axis, Camel, BeanUtils, Collections, Code, Discovery, HTTP Client, Lang, Math, VFS, Hazlecast, Jakarta, Log4j, Jakarta ORO, Jasper, Lucense, Regexp, Velocity, XML-RPC, Axis, Commons, Felix, Expression Language
LinkedIn Mobile apps https://www.linkedin.com/legal/mobile/open-source-mobile-apps	ActionBar Sherlock, Apache HTTP, Butterknife, Commons IO, Cropper Edge, Cropper Image View Utility, Cropper Paint Utility, dagger, Diff-patch-map, EventBus, Google Diff Match Patch, Google Authenticator's PAM module, Google Toolbox for Mac, Gson, Hakawai, Jackson parser, JSONkit, KIF, List View Expanding Cells, OAuth Consumer, Ocmock, OkHttp, Okio, Oto, PageSlidingTabStrip, PhotoView, Pinned Section ListView, Retrofit, Seismic, Selene, Sliding menu, Sliding Tabs Colors, SnappyDB, SuperToolsTips
Drobox iOS app	Apache-Jakarta HTTP Client, ASAccessibilityElement, dropbox-notes, google-toolbox-for-mac, HttpCore – httpcomponents-httpcore:Jakarta-httpcore, ios-i18n, Jakarta Commons-Logging, json-simple, parsekit, three20UICommon-facebook.
Evernote client application https://evernote.com/opensource/	Thrift, log4net
Instagram client application https://www.instagram.com/about/legal/libraries/	Thrift, QSUtilities, SocketRocket,

Epson Expression XP-211 https://files.support.epson.com/htmldocs/xp211_/xp211_ug/Source/Printers/Source/Notices/Open_Source/oss_terms_fy13_package_6.html	Bonjour

47. Selecting the open-source license for an open-source project (the "project license") can impact outside participation in a project and how the project software will be combined with third-party open-source and closed-source projects and products. Google selected the Apache-2.0 license as the preferred license for the Android project. Google's licensing notice provides that, unless software was specifically designated by Google as made available under an open-source license *other than* the Apache-2.0 license, the Apache-2.0 license applies to Android projects sponsored by Google. Certain software packages included in the Android platform incorporate third-party software that is subject to strong-copyleft or weak-copyleft licenses. Those packages are distributed under the same strong-copyleft or weak-copyleft license. The Linux kernel included in the Android platform, for example, is licensed under the GPL-2.0 license.

48. In response to inquiries regarding the choice of the Apache-2.0 license as the preferred Android license, Google published the statement copied below at https://source.android.com/source/licenses.html. As explained in the statement, Google selected the permissive Apache-2.0 license over a version of the weak-copyleft LGPL license to enable broader, less-encumbered use of the Android software on mobile devices. Google identified two specific obligations in the LGPL license that cause challenges or concerns for commercial software and commercial device manufacturers: (a) the obligation to enable downstream recipients of software linked with an LGPL library to replace that library (standalone or as

embedded on device) with their own preferred version; and (b) the obligation to permit downstream modification and reverse engineering of LGPL libraries.

> We are sometimes asked why Apache Software License 2.0 is the preferred license for Android. For userspace (that is, non-kernel) software, we do in fact prefer ASL2.0 (and similar licenses like BSD, MIT, etc.) over other licenses such as LGPL.
>
> Android is about freedom and choice. The purpose of Android is promote openness in the mobile world, and we don't believe it's possible to predict or dictate all the uses to which people will want to put our software. So, while we encourage everyone to make devices that are open and modifiable, we don't believe it is our place to force them to do so. Using LGPL libraries would often force them to do just that.
>
> Here are some of our specific concerns:
>
> - LGPL (in simplified terms) requires either: shipping of source to the application; a written offer for source; or linking the LGPL-ed library dynamically and allowing users to manually upgrade or replace the library. Since Android software is typically shipped in the form of a static system image, complying with these requirements ends up restricting OEMs' designs. (For instance, it's difficult for a user to replace a library on read-only flash storage.)
>
> - LGPL requires allowance of customer modification and reverse engineering for debugging those modifications. Most device makers do not want to have to be bound by these terms. So to minimize the burden on these companies, we minimize usage of LGPL software in userspace.
>
> - Historically, LGPL libraries have been the source of a large number of compliance problems for downstream device makers and application developers. Educating engineers on these issues is difficult and slow-going, unfortunately. It's critical to Android's success that it be as easy as possible for device makers to comply with the licenses. Given the difficulties with complying with LGPL in the past, it is most prudent to simply not use LGPL libraries if we can avoid it.
>
> The issues discussed above are our reasons for preferring ASL2.0 for our own code. They aren't criticisms of LGPL or other licenses. We are passionate about this topic, even to the point where we've gone out of our way to make sure as much code as possible is ASL2.0 licensed. However, we love all free and open source licenses, and respect others'

> opinions and preferences. We've simply decided ASL2.0 is the right license for our goals.

Android Licenses published at https://source.android.com/source/licenses.html

49. As discussed in greater detail below, linking (dynamically or statically) with software licensed under the GPL-2.0-CE license does not raise the same challenges and concerns Google raised with respect to the LGPL license.

F. The GNU Licenses

50. The FSF is the curator of the "Free Software Definition" as well as the GNU licenses which, to varying degrees, embody the copyleft philosophy of the FSF. The most commonly used GNU licenses are the GPL and LGPL licenses. There are multiple published versions of each license.[9] While the specific obligations and restrictions imposed varies by version for each license, the copyleft effect imposed by those versions have comparable copyleft scope. For example, the LGPL-2.0, LGPL-2.1, and LGPL-3.0 licenses each explicitly allow at least some form of linking with licensed LGPL libraries without imposing a copyleft effect on the linking software. The GPL-1.0, GPL-2.0, and GPL-3.0 licenses each extend copyleft effects to distributed derivative works of the GPL-licensed software.

51. Providers of closed-source products and services and differently licensed open-source projects often perceive the GPL to be more burdensome because the broader perceived copyleft effect may impose copyleft requirements on combined software that would not be imposed under the LGPL.

[9] The AGPL-3.0 license is the only version of the AGPL published by the FSF. A previous version of the license was published by an independent organization.

1. The GNU General Public License

52. The FSF published the GPL-2.0 license in 1991 and it remains the most commonly used version of the GPL.[10]

53. The table below identifies popular open-source projects licensed under the GPL-2.0 license as well as some of the companies that own, sponsor, or contribute to those projects:

Project	**Companies owning, sponsoring, or contributing to the project**
Linux https://www.kernel.org/	Intel, Red Hat, Samsung, IBM, Texas Instruments, Google, Renesas Electronics, Oracle, AMD, NVidia, Broadcom, Huawei, ARM, Cisco, Qualcomm, and Fujitsu.
Drupal https://www.drupal.org/	Facebook, NBC Universal, Home Depot, Pfizer, MLSSoccer.com, Red Hat, Hewlett-Packard, Tesla Motors, Qualcomm,

54. The table below identifies commercial products and services that incorporate software licensed under the GPL-2.0 or GPL-3.0 license:

[10] Published at https://www.gnu.org/licenses/old-licenses/GPL-2.0 license.html.

Product	**GPL packages**
Oracle Integrated Lights Out Manager (ILOM) https://docs.oracle.com/cd/E56047_01/html/E24527/z40001041148957.html	BusyBox, Das U-Boot, adduser, apt, apt-utils, aptitude, bas-files, base-passwd, bash, coreutils, cpio, cpp, debianutils, defoma, diff, discover, dmidcode, dosfstools, dpkg, dselect, e2fslibs, e2fsprogs, eject, ethtool, findutils, fvwm, grep, gzip, hostname, ifupdown, iptables, initscripts, klogd, mawk, module-init-tools, mount, net-tools, netbase, nfs-common, pciutils, sed, slang, ssh, sysklogd, sysv-rc, sysvinit, tar, ucf, vnc-common, vncserver, wget
Oracle Endeca Commerce (Sept 2013, January 2014)	Makeself, ndoc, EndecaRoot utilities
Oracle Big Data Appliance http://docs.oracle.com/cd/E55905_01/doc.40/e55807/thirdparty.htm#BIGLI116 https://docs.oracle.com/cd/E65728_01/doc.43/e65667/GUID-E4C79F9B-21A9-4EF1-B643-6BA3638022F5.htm#BIGLI116	Linux, Puppet
Cisco TelePresence Software TC6.0 https://cisco-images.test.edgekey.net/c/dam/en/us/td/docs/telepresence/endpoint/software/tc6/licensing_guide/sx20_open_source_documentation_tc60.pdf	alsa-lib_aserver, alsa-utils, binutils, bootcharts, coreutils, ethtool, fbset, findutils, gawk, gcc, GFX Linux KM, grep, gzip, inotify-tools, iproute2, iptables, iputils, linux, module-init-tools, mtr, nc6, ndisc6, net-tools, netcat, nfs-utils, procps, readline, rsync, schedutils, sed, sysvinit, tar, termcap, udev, util-linux, wpa_supplicant, yaffs2

Cisco SPA112, SPA 122, SPA232D, and SPA302D telephone system products http://www.cisco.com/c/dam/en/us/td/docs/voice_ip_comm/csbpvga/spa100-200/release/SPA112_SPA122_SPA232D_SPA302D_1_3_2_Firmware_Open_Source_Documentation.pdf	Bridge, busybox, clean_net_count, dnsmasq, ebtables, iproute2, iptables, led, linux kernel, linux-cordless-driver, makedevs, mtd-utils, net-tools, netfilter, ntpclient, squashfs, sysevent, syslog-ng, u-boot bootloader, u-boot_env, udhcp,
Sony BDP-S1200, BSD-S3200, BSD-4200, BDP-S4200, BDP-S7200, BDP-BX120, BDP-BX320, BDP-BX520 and BDP-BX620 BluRay players. http://oss.sony.net/Products/Linux/Video/BDP-S1200.html	Alas-lib, busybox, coreutils, e2fsprogs, iproute2, iputils, net-tools, linux, procps, u-boot
Cisco Digital Media Player 4400G http://www.cisco.com/c/en/us/td/docs/video/digital_media_systems/dmp/open/source/4400G-5-2.html	BusyBox, linux
Cisco CSC SSM Administrator http://www.cisco.com/c/en/us/td/docs/security/csc/csc63/administration/guide/csc_admin/cscappe.html	Bash, binutils, busybox, diffutils, e2fsprogs, grub, iptables, kysmoops, Linux-PAM, mod-utils, procfs, syslog-ng, systtat, termcap, util-linux, Linux kernel
GoPro Hero3+ and Hero3 cameras https://gopro.com/support/open-source	Autconf, automake, bandwidth, busybox, Cherokee, dsnmasq, e2fsprogs, fakeroot, libslack, tzo, module-ini-tools, mtd-utils, wireless_tools, Linux kernel.

2. The GNU Library/Lesser General Public License

55. The LGPL license was first used with GNU's C library (which is linked with C programs compiled by the GNU C compiler). FSF founder Richard Stallman, who selected the LGPL-2.0 license for the C library, considered using the GPL-2.0 license instead. However, under the FSF's published interpretations of the GPL-2.0 license,[11] a GPL-licensed C library would result in all programs compiled with the GNU C compiler being subject to the GPL-2.0 license's copyleft effect (*aka* being "GPL'd" or "tainted"). Stallman was concerned that selecting

[11] See *Frequently Asked Questions about the GNU Licenses*, published at https://www.gnu.org/licenses/gpl-faq.html.

the GPL-2.0 license would prevent development and distribution of closed-source programs using the GNU C Compiler and, in turn, discourage the adoption of GNU/Linux platform. Accordingly, the FSF published the GNU C library under the newly introduced LGPL-2.0 license, which permitted linking software licensed under LGPL-incompatible open-source and closed-source licenses, without imposing copyleft obligations on the linking software.[12]

3. The Classpath Exception

56. The "Classpath Exception" to the GPL was created by the FSF to facilitate the GNU Classpath project. This project was founded to provide an open-source alternative to the standard Java libraries, which at the time were made available by Sun (Oracle's predecessor-in-interest for Java) under a limited license that was neither a free software nor open-source license. (These libraries were later released under open-source licensing terms via the OpenJDK project, discussed below.) The Classpath Exception was necessary because the FSF took the position that, without it, the standard libraries could not be combined with closed-source programs. Under the FSF's published interpretation of the GPL-2.0 license, linking libraries licensed under the GPL license with closed-source and other GPL-incompatible licenses would violate the GPL license and could result in all software compiled with the GNU Compiler being subject to the GPL-2.0 license's copyleft effect (being "GPL'd" or "tainted"). The FSF also viewed the additional permissions of the LGPL to be insufficient to promote broad use of the Classpath libraries. Despite not imposing a copyleft effect on linked software, the LGPL includes certain compliance obligations that are at least inconvenient to closed-source product distributors and presented significant challenges and distribution deterrents for manufactures and distributors of

[12] See Richard Stallman, *The GNU Project*, published at https://www.gnu.org/gnu/thegnuproject.html; GNU.org, *Why you shouldn't use the Lesser GPL for your next library*, published at http://www.gnu.org/licenses/why-not-lgpl.html.

devices (such as a televisions, mobile phones, and kitchen appliances) including embedded LGPL-licensed software.

57. To encourage use of the libraries by developers of both GPL-compatible and GPL-incompatible software, the GNU Classpath project amended the GPL-2.0 license to include a "Classpath Exception," which excused distributors of GPL-incompatible software merely linking to Classpath Exception software from not just the GPL-2.0 license's copyleft effects but from all compliance requirements relating to the GPL-2.0-CE licensed software:

> Linking this library statically or dynamically with other modules is making a combined work based on this library. Thus, the terms and conditions of the GNU General Public License cover the whole combination.
>
> As a special exception, the copyright holders of this library give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you may extend this exception to your version of the library, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

GNU Classpath License published at: https://www.gnu.org/software/classpath/license.html.

58. In November 2006, Sun announced its intention to distribute a reference implementation the sixth Java specification (Java SE 6) under a GPL-compatible license. In 2007, Sun released the JVM under the GPL-2.0 license and the core Java libraries under the GPL-2.0-CE license, the same license chosen by the FSF for the GNU Classpath project charged with replacing those libraries. Soon after Sun's announcement in 2006, the FSF published a post lauding Sun's decision to "[begin] disarming" what FSF founder and President Richard Stallman called the "Java Trap."[13]

[13] Post published at: http://www.fsf.org/news/fsf-welcomes-gpl-java.html.

59. Sun published an FAQ regarding its release of Java under an open-source license and its selection of the GPL-2.0-CE license (the "Java FAQ").[14] The FAQ described the GPL-2.0-CE license and its selection for the core Java libraries:

> Q: What license did you choose for the open-source JDK components?
>
> A: GPL v2 for almost all of the virtual machine, and GPL v2 + the Classpath exception for the class libraries and those parts of the virtual machine that expose public APIs.
>
> Q: What is the Classpath exception?
>
> A: The Classpath exception was developed by the Free Software Foundation's GNU/Classpath Project (see http://www.gnu.org/software/classpath/license.html). It allows you to link an application available under any license to a library that is part of software licensed under GPL v2, without that application being subject to the GPL's requirement to be itself offered to the public under the GPL.
>
> Q: Why did you choose this licensing method?
>
> A: This is the licensing paradigm in common use within Free software communities such as GNU/Classpath and Kaffe for the components of a Java technology implementation including the virtual machine and class libraries. We consciously chose the same licensing method so that there would be no temptation to second guess Sun's intention to make its Java SE implementation available under a genuinely Free and open license and to allow easy collaboration with these existing communities.
>
> Q: Doesn't GPL v2 + Classpath exception offer very similar licensing terms to the LGPL? Why not use LGPL instead?
>
> A: Yes, from a practical perspective the Classpath exception establishes similar terms to LGPL. However, the Free software Java technology communities haven't chosen to use LGPL, and so we at Sun decided to follow their lead and use the Classpath exception.

60. The GPL-2.0-CE license is a weak-copyleft license. Like other weak-copyleft licenses, the GPL-2.0-CE license has a copyleft effect on certain modifications to the licensed

[14] The Java FAQ site originally posted by Sun has been taken down and replaced since Sun's acquisition by Oracle. The original Java FAQ is still published in the OpenJDK section of the IcedTea project and is published available at http://icedtea.classpath.org/openjdk/java/faq.jsp.html. A snapshot of the original Java FAQ site taken June 4, 2011, by Archive.org is available at: https://web.archive.org/web/20110604004915/http://www.sun.com/software/opensource/java/faq.jsp. Contemporaneous versions were also produced in this case as GOOG-00000221 and GOOG-00000316.

software but not linking software. Specifically, the GPL-2.0-CE license requires that distributed modifications to GPL-2.0-CE-licensed libraries also be licensed under the GPL-2.0-CE license.[15] Like many other weak-copyleft licenses, the GPL-2.0-CE license also explicitly permits linking with GPL-2.0-CE-licensed software without imposing a copyleft effect on the linking software. Sun reached the same conclusion regarding classification of the GPL-2.0-CE license in its Java FAQ stating that "from a practical perspective the Classpath exception establishes similar terms to LGPL."[16]

61. The GPL-2.0-CE license is distinguishable from many other weak-copyleft licenses in that the GPL-2.0-CE license exempts those qualifying for the Classpath Exception from any compliance obligations related to the GPL-2.0-CE-licensed software. In fact, the only obligation placed on distributors qualifying for the Classpath Exception is compliance with the requirements imposed by whichever other license(s) may be covering the *linking* GPL-incompatible software. By contrast, many other weak-copyleft licenses (such as the LGPL, MPL, and CDDL) obligate software distributors to offer or provide recipients with the source code corresponding to the distributed weak-copyleft packages under the same license. Weak-copyleft licenses often obligate distributors to provide recipients with additional notices such as warranty disclaimers, a copy of the applicable weak-copyleft license, or other licensing information. By releasing qualifying software distributors from any obligation with respect to the GPL-2.0-CE-licensed software, the GPL-2.0-CE license places fewer obligations on qualifying distributors than many permissive open-source licenses.

62. In its Classpath FAQ, the GNU Foundation described the effect of the GPL-2.0-CE license's Classpath Exception as follows:

[15] The GPL-2.0-CE also permits distribution under the GPL-2.0 without the additional Classpath Exception.
[16] Java FAQ, *supra.*

> What does the exception allow me to do?
>
> If you combine GNU Classpath with independent modules to produce an executable you can copy and distribute the resulting executable under terms of your choice. So you can use and distribute GNU Classpath as is in your program without changing the license of your software.

63. Sun acknowledged the possibility that its releasing Java platform components under the GPL-2.0 and GPL-2.0-CE licenses could lead to Java implementations that do not comply with the Java specification and the reuse of only certain packages included under GPL-2.0-CE in its Java FAQ:[17]

> Q: Can someone create and distribute an implementation that isn't compatible with the Java specification using this code?
>
> A: Yes. We do not recommend or endorse that action, however. In addition, they cannot label that implementation with the Java Compatible or Java Powered (for Java ME) brand and logo. These brands are your assurance that an implementation has passed the relevant TCKs.
>
> Q: Can someone create software that doesn't even implement Java, but uses pieces of the OpenJDK code commons? What are the limitations, if any?
>
> A: Yes. There are no limitations. But there is an obligation to meet the requirements of the GPL (plus Classpath and Assembly exceptions if appropriate).

GOOG-00000221 (at -240); GOOG-00000316 (at -335).

64. Sun also acknowledged in its Java FAQ[18] that the release of the Java Platform under the GPL-2.0 license and GPL-2.0-CE license could lead to Java implementations on platforms not supported by Sun:

> Q: What impact do you see open sourcing the JDK will have on bringing Java to platforms that Sun may not be directly interested in supporting? For instance, could the community provide an implementation for a game console? Can a bunch of like-minded engineers simply approach the console manufacturer and offer to compile a JVM for them for free from the OpenJDK sources?

[17] Java FAQ, *supra*.
[18] Java FAQ, *supra*.

> A: Great question - because we think one of the most important benefits of open-sourcing Sun's platform implementations will be the new platforms that will be supported by the community. And to your point - once the complete JDK is available under the GPL, it will be free software, and developers will be able to use it in any way they see fit, as long as they abide by the GPL, including publishing any modifications to the platform they might make, benefiting the whole Java technology ecosystem. We think there are many opportunities to bring Java to new platforms, and neither licensing nor technology really stands in the way in most cases.

GOOG-00000221 (at -253); GOOG-00000316 (at -348).

65. Sun also noted in its Java FAQ[19] that certain open-source license compliance requirements stemming from the licensing of the JRE under the GPL-2.0 license could entice commercial hardware distributors to purchase commercial Java licenses from Sun:

> That said, there are significant issues with delivering a JRE embedded in a platform such as a game console. The most significant issue is that if the JRE built from GPLed code is distributed by the manufacturer as an integral component of their game system, the entire source code for the game system would need to be made available under the GPL as well. Revealing the proprietary APIs and implementation details of the game system might not be what the manufacturer would wish for. In this case, if they see value in using Java technology to make their game platform more open to application developers, they would likely want to do so under a commercial license from Sun, which would protect their interests.

GOOG-00000221 (at -253); GOOG-00000316 (at -348).

66. The GPL-2.0-CE license has become a popular license choice among open-source projects producing distributable Java libraries, plug-ins, and other platform components. Examples of open-source Java projects adopting the GPL-2.0-CE license include NetBeans, Grizzly, JavaBeans Activation Framework (JAF), JavaHelp, JavaMail, JAXP, Jersey, Metro, SAAJ, GNU Classpath, IcedTea, Java API for JSON Processing RI, Graphene, and GlassFish

[19] Java FAQ, *supra*.

(sponsored by Oracle). NetBeans is an Oracle-sponsored open-source integrated development environment (IDE) for Java also licensed under the GPL-2.0-CE license. As of the preparation of this report, a NetBeans FAQ titled "Why GPL v2"[20] continues to mirror the description of the Classpath Exception provided in the Java FAQ.

67. Applications executing as separate processes and communicating with GPL-2.0-licensed software via only IPC don't need to qualify for the Classpath Exception to avoid the GPL-2.0-CE license's copyleft effect. Indeed, where IPC is the only means of communication and data sharing between two separately executing processes, the two processes would almost certainly not qualify as derivative works of one another and the combination would qualify for even the strong-copyleft GPL-2.0 license's aggregation exception explicitly excusing the communicating software from the GPL-2.0 license's copyleft effects when distributed in combination with the GPL-2.0-licensed software with which it communicates.[21]

68. Even the FSF, which consistently takes a broad reading of the GPL-2.0 license's copyleft effect, reaches the same general conclusion. Specifically, the FSF contends in its GPL FAQ that, in most instances, IPC commonly used between separate programs, such as pipes, sockets, and command-line arguments, will not give rise to the creation of a derivative work.[22]

[20] See NetBeans *Why GPL v2 Frequently Asked Questions* published at https://netbeans.org/gplv2-faqs.html.

[21] See, for example, Lothar Determann, *Dangerous Liaisons – Software Combinations as Derivative Works? Distribution, Installation, and Execution of Linked Programs Under Copyright Law, Commercial Licenses and the GPL*, 21 Berkeley Technology Law Journal 1421, 1450 (2006) ("All these IPC mechanisms. . . remain separate and distinguishable in the logical program view. The selection of the most appropriate IPC is dictated by external functionality requirements … Hence, programs combined like this would generally not be considered part of a larger derivative work due to a lack of combination creativity and significant internal changes."); *Ron Phillips, Deadly Combinations: A Framework for Analyzing the GPL's Viral Effect, 25 J. Marshall J. Computer & Info. L. 487, 501 (2008)* (concluding that "[p]rograms that communicate through some form of inter-process communication ("IPC") such as sockets and named pipes conform to the interface and data standards imposed by those IPC mechanisms" and "Interoperability that is based on reading and writing files stored on a disk probably [do] not constitute a derivative work.").

[22] GNU.org, GPL FAQ, #MereAggregation ("By contrast, pipes, sockets, and command-line arguments are communication mechanisms normally used between two separate programs. So when they are used for communication, the modules normally are separate programs.")

69. Because the GPL-2.0 license limits its copyleft effect to distributed derivative works of the GPL-2.0-licensed software and separately executing processes are highly unlikely to be considered derivative works of another, software communicating with GPL-2.0-CE licensed software only through one or more forms of IPC would not be subject to the GPL-2.0-CE license's copyleft effect, regardless of whether the Classpath Exception applies.

G. Business Models Related to Open-Source Software

70. Companies sometimes choose to publish their software under the terms of an open-source license (often referred to as "open-sourcing" the software) for many different reasons including: (a) to increase adoption of company products or platforms for or through which the company sells add-ons, plug-ins, content or support services; (b) to reduce engineering costs through crowd-sourced and cooperative development and testing or otherwise externalizing company development costs; (c) to create goodwill within the open-source and business communities; (d) to increase company knowledge, experience, and expertise with technology and software on which the company depends or that the company is considering adopting; (e) to increase influence over the development roadmap of the open-source project on which the company depends; and (f) as part of open-source business model.

71. The most commonly adopted open-source business models include one or more of the following: (a) providing products and services that complement open-source projects; (b) offering commercially friendly licensing terms for software otherwise available on open-source terms; (c) providing limited versions of the software under an open-source license and commercially licensing enhanced versions, plugins, or extensions to the open-source software; (d) providing an open-source platform and commercially licensing enhanced versions, plugins, extension or applications for the platform or charging a royalty for software or content distributed via the platform; and (e) providing enhanced closed-source distributions of popular

open-source projects. These business models are described in greater detail in the remainder of this section.

1. Offering Complementary Products and Services

72. The most commonly used open-source business model offers products, services, or combinations thereof that complement or otherwise support popular open-source projects. The plethora of such supporting products and services can be explained, in part, by the ability of nearly anyone to offer competitive products and services for the same open-source project. Unlike some of the business models described in the remainder of this report, providing supporting products and services does not necessarily require that the provider have exclusive rights to the delivered product.[23] The provider's lack of exclusive rights may result from the provider not generating significant software or other IP in the course of providing its products or services or because the IP that is generated is either distributed to recipients under an open-source license or contributed back to the open-source project complemented by the products or services.

73. Examples of supporting services include customizing the open-source project to meet a customer's unique needs; providing support, maintenance, development, consulting, or training services to customers that depend upon a particular open-source project; and providing auditing and legal services relating to open source. For example, Red Hat, Canonical, Novell, and others offer support, maintenance, design, and consulting services relating to their respective custom Linux distributions (e.g., RHEL, Ubuntu, and Suse). Pivotal similarly provides consulting, development, and training services related to its open-source software development platform, Spring. Companies like Black Duck, Palamida, and Protecode offer source code audit

[23] Exclusive rights may include exclusive copyrights, patent rights, and trade secrets.

services that identify specific open-source packages included in commercial software distributions.

74. Examples of supporting products include hosting services for common combinations of open-source packages (often referred to as "stacks"),[24] providing warranty and indemnification coverage for open-source use, offering early access to updates and other software that will eventually be released under an open-source license, and the licensing of software products that support the commercial use of open-source software. MongoLab, for example, provides hosting services for the popular MongoDB open-source database project. MongoDB Inc. offers commercial licenses that can include licensee indemnification for claims made against the licensee based on its use of the MongoDB database software. RedHat likewise offers its "intellectual property assurance program" to paid Red Hat subscribers. Since acquiring Sourcefire, Cisco has continued the practice of granting paid subscribers early access to rule sets later released on a royalty-free basis for use with its open-source-licensed network security software, Snort. Palamida and Black Duck both offer governance software tools for scanning and managing open-source use within a company.

2. Dual Licensing

75. Another frequently adopted open-source business model offers the same software under either (a) an open-source license, or (b) a more traditional commercial license obviating the need to comply with obligations imposed by the open-source license. Unlike the open-source products and services described in the previous section, companies adopting this "dual-licensing" or "multi-licensing" approach generally own or have some exclusive rights to license the

[24] Software functionality delivered via the internet is often referred as providing software "as a service" (SaaS). Providers that purchase, host, support, maintain, and lease to customers the servers and other hardware necessary to run such software are often said to be providing hosting services. While commonly described as services, the primary resource consumed in providing *additional* services are technical (e.g., additional servers and networking equipment), not human. Accordingly, for the purposes of this Report such software and hosting services are considered products rather than services.

software offered under multiple licenses. The open-source license offered by the license is often a strong-copyleft license or a prohibitive public-source license with obligations inconsistent with at least some closed-source and/or commercial uses of the software. Often, the intended use of the dual-licensed software will arguably have a copyleft effect on distributors' or hosted users' proprietary software. The copyright holder offers a commercial license to companies that wish to use or distribute the software without being subject to the undesirable requirements of the open-source license. Alternatively, some commercial licenses may provide access to additional products and services such as those described in the previous section.

76. Examples of dual-licensed software with open-source and traditional commercial licensing alternatives include Oracle's MySQL (GPL-2.0 license), Berkeley DB (AGPL-3.0 license), and Oracle JDK (BCLA license), Digium's Asterisk (GPL-2.0 license), and Qt (LGPL-2.1 license).

77. Some dual-licensed software is provided under either of two or more open-source licenses for licensing convenience. The NetBeans IDE, for example, is offered under *either* the CDDL-1.0 license or the GPL-2.0-CE license.

3. Open Core/Freemium

78. Some companies offer "standard," or "lite" or similarly named limited versions of their software under an open-source license, while offering "enterprise" or similarly named enhanced versions of the software under more traditional commercial-licensing terms. Commercial versions of the licensed software typically include enhanced functionality or performance particularly desirable to a subset of the project's users whom the licensor hopes to convert to commercial users. So-called "open-core" or "freemium" software offerings are not unlike the service models adopted by providers such as Dropbox, Pandora, LinkedIn, Evernote,

and MailChimp. Examples of open-core software with enhanced commercial counterparts include Proofpoint's Sendmail, SugarCRM's Sugar, and Zimbra's Collaboration software.

4. Open Platform

79. Some companies provide a computing or service platform under an open-source license. Like open core, open-platform providers often have commercial products, such as plug-ins or extensions, which complement open-source platforms owned or contributed to by the provider. Some platform providers also enable the delivery and use of commercial applications, services, or content subject to a surcharge collected by the platform provider. The Eclipse software development platform, for example, was originally released under an open-source license by IBM, which still (along with many others) sells commercial plugins for Eclipse Foundation's platform. Pentaho licenses its core business analytics platform under the GPL-2.0 license while offering commercial plugins and extensions authored by Pentaho and others through its Pentaho Marketplace.

5. Closed-source distribution of open-source software

80. In its broadest sense, closed-source use and distribution of open-source software may represent the broadest use of open-source software by commercial enterprises. In fact, nearly all closed-source software products include at least some open-source software. However, with respect to open-source business models, closed-source distribution of open-source software refers to companies selling closed-source versions of popular open-source projects. Permissively licensed open-source can be modified and combined with the provider's proprietary software without obligating the provider to give access to the source code for the provider's proprietary code. Companies such as Cloudera, for example, offer a commercial version of the permissively licensed Apache Hadoop project software.

H. Combining Open-Source Software

81. A number of software platforms and stacks have also been released under one or more open-source licenses. It is not uncommon for the different components of such a platform to be released under different open-source licenses and, in some instances, even commercial licensing terms. The choice of license for a particular component often depends on how the components will be included and used in the platform and combined with programs developed for use with the platform. The table below includes examples of open-source programming languages that have been released and use software contributed or licensed to the project under the terms of multiple open-source licenses.

Programming language	Major project components include	Component licenses	Included code licenses
OpenJDK http://openjdk.java.net/	• Java compiler • Java virtual machine • Java class libraries	• GPL-2.0 license • GPL-2.0-CE license • GPL w/ assembly exception	• LGPL-2.1 • MPL-1.1/2.0 • Apache-1.1/2.0 • BSD • MIT • W3C • zlib/libpng • Unicode
GNU C Compiler (GCC) https://gcc.gnu.org/	• C compiler • Runtime libraries	• GPL-2.0 license • LGPL-2.1 • GPL-2.0 license with GCC Runtime Library Exception	• BSD • MIT
GNU Classpath http://www.gnu.org/software/classpath/license.html	• Java compiler • Java class libraries	• GPL-2.0 license • GPL-2.0 license with Classpath Exception (GPL-2.0-CE license) • MIT	• BSD • MIT
GNU GNAT http://www.gnu.org/software/gnat/	• Ada95 compiler • Runtime libraries	• GPL-3.0 • LGPL-3.0	
GNU Guile http://www.gnu.org/software/guile/	• Scheme interpreter • Application framework • Plugin modules and libraries	• GPL-3.0 • LGPL-3.0 • LGPL-2.1+	
Xamarin Mono https://xamarin.com/	• .NET runtime • .NET core libraries • .NET runtime libraries	• GPL-2.0 license • LGPL-2.0 license • MIT	• Apache/MPL • MIT • Ms-PL • Zlib/libpng
Free Pascal http://freepascal.org/	• Pascal compiler • Runtime libraries	• GPL-2.0 license • LGPL-2.0 license • GPL-2.0 license with linking exception	• Apache-2.0 license • BSD

82. Combinations of software provided under different open-source software licenses can raise issues of "license compatibility." When the requirements imposed by two or more open-source licenses can always be satisfied simultaneously, those licenses are said to be

"compatible." That is, when a specific combination of software is subject to multiple open-source licenses and the requirements of those licenses can be satisfied simultaneously for that combination, those licenses are said to be compatible with respect to that combination. And when a specific combination of software is subject to multiple open-source licenses but the requirements of those licenses cannot be satisfied simultaneously, then the licenses are said to be "incompatible" with respect to that software combination. License incompatibility may result when a certain combination of software includes portions licensed under one or more copyleft licenses in impermissible combinations with software licensed under an incompatible license. Incompatibility may also result when an obligation or restriction imposed by one applicable open-source license conflicts with the obligations or restriction of another open-source license.

83. Many open-source projects include components, packages, or code "snippets" contributed to or licensed for use in the project under terms that differ from the project license. Some projects release different components of the project software under different open-source or, in some cases, even commercial licensing terms. The table below includes a few examples of open-source projects that distribute different components under distinct licenses or include software packages licensed to the project under terms that differ from the project license.

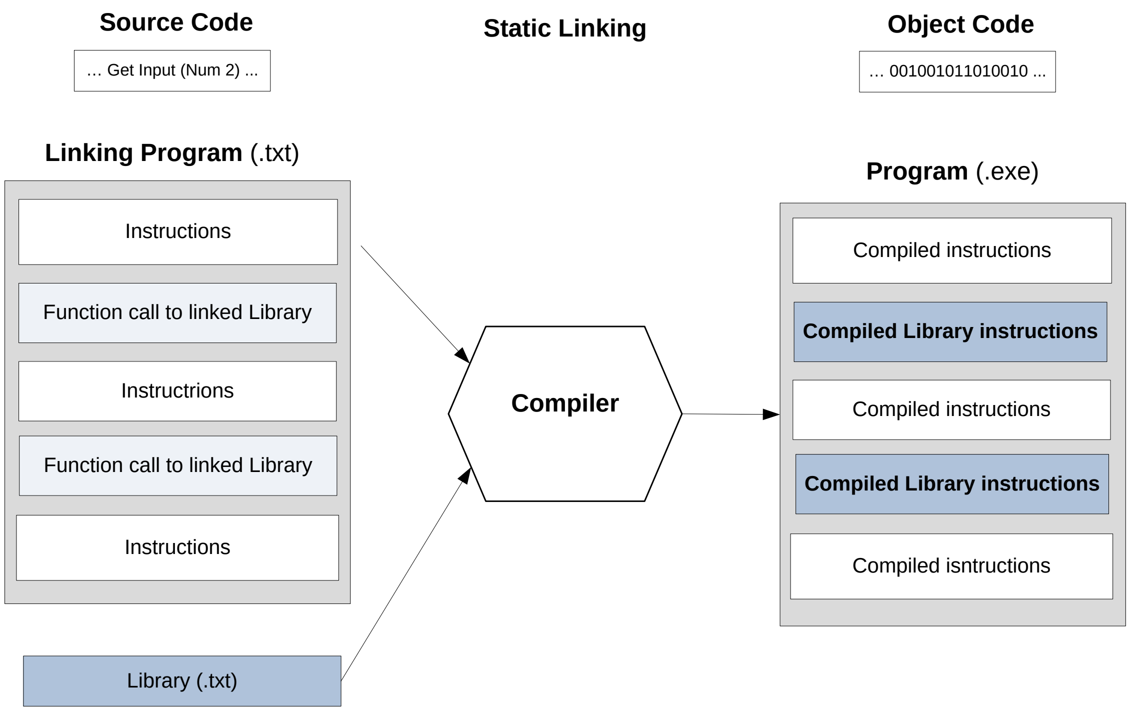
Project	Major components include	Component licenses	Included code licenses
Linux https://www.kernel.org/	• Linux kernel; • Linux kernel modules (drivers); and • Bundled Linux apps, tools, and middleware.	• GPL-2.0 license • LGPL-2.1+	• BSD • MIT
Docker https://www.docker.com/	Docker engine	Apache-2.0 license	• BSD • MIT
VLC Media Player https://www.videolan.org/vlc/	• Media player • Interface library	• GPL-2.0 license • LGPL-2.1	• BSD • MIT
LibreOffice https://www.libreoffice.org/	• Office applications • Application framework	• MPL-2.0 / LGPL-3.0+	• LGPL-2.1/3.0 • MPL-1.1 • Apache-2.0 license • Boost-1.0 • MIT • W3C

I. Software Linking

84. **Linking.** The term "linking" is used generically by software developers to describe various software combinations. The term may refer to the combination of two or more separate packages of software into a single program. Often, those packages consist of the main program code and one or more software libraries providing certain well-defined and limited functionality (such as data encryption or PDF manipulation) to the "linking" main program.

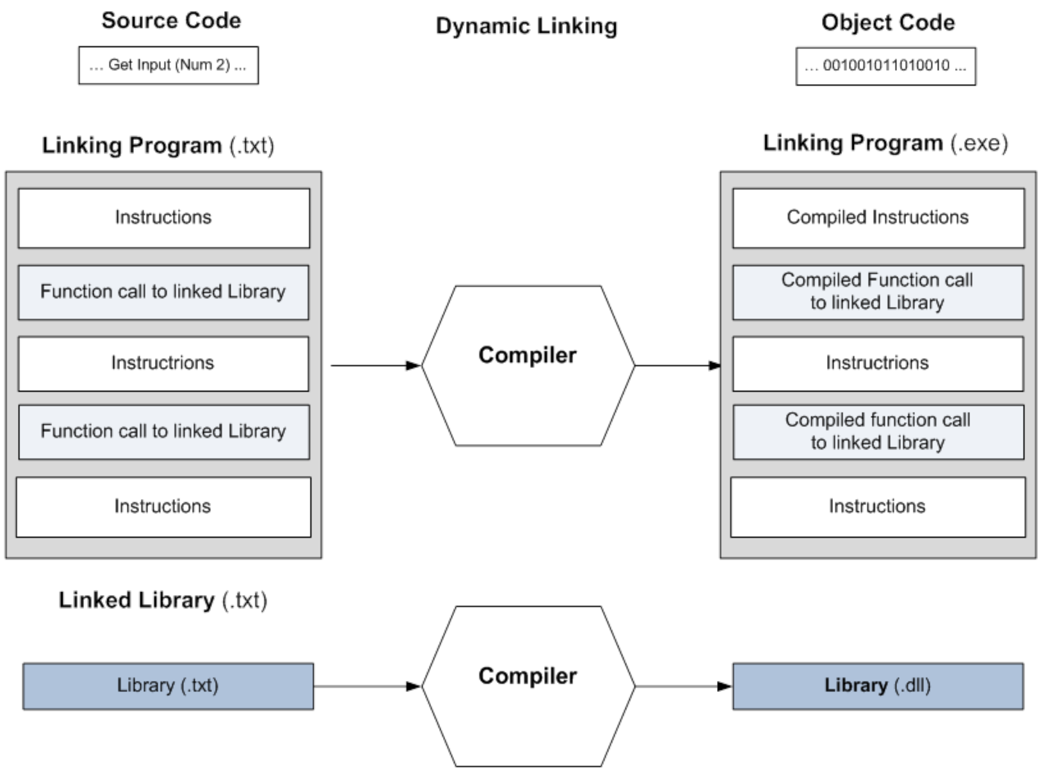
85. Almost all software is developed in modular pieces or packages. For example, if a programmer wants to accomplish a common task (such as printing a file to a printer or getting input from a keyboard), it is not efficient for the programmer to write software to accomplish that task each time the developer writes a program. Instead, the programmer for the main program can simply refer to existing software libraries when writing the source code for the main program. These software elements are combined to form a working program by a process called linking. The linking process then produces an executable binary program using both the main program and the libraries upon which it depends.

86. **Static Linking.** Programmers often distinguish between two types of linking: static (or compile-time) and dynamic (or runtime) linking. When a library is statically linked to another program, the compiling[25] and linking processes combine the main program code and the referenced library code into a single "monolithic" executable file as depicted in the figure below.

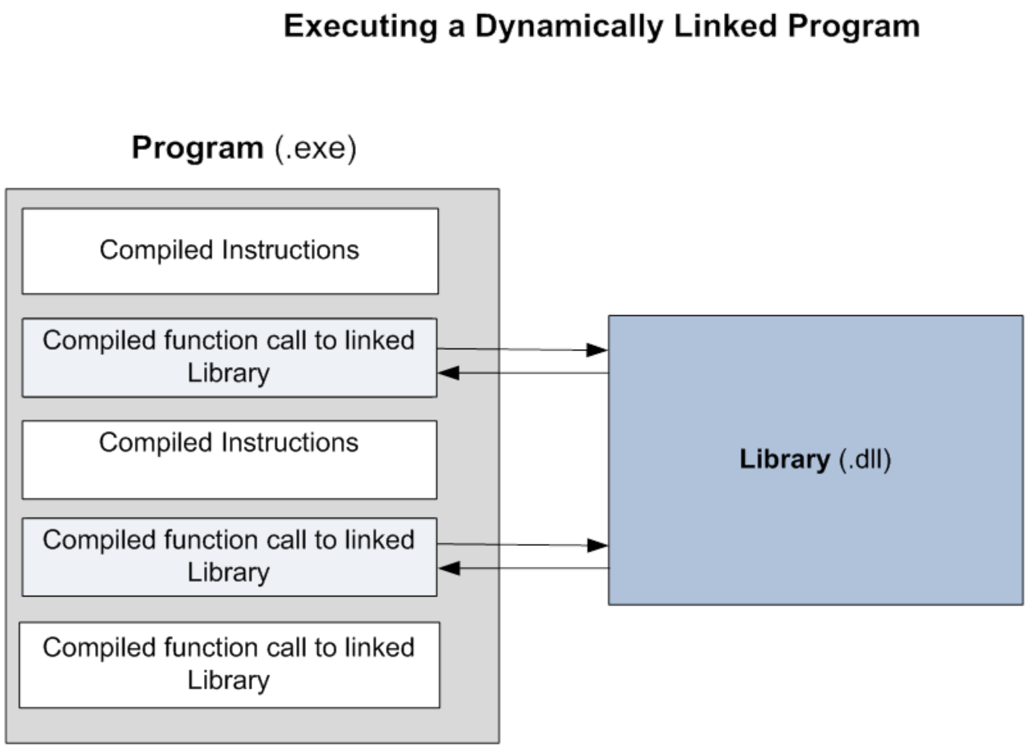


87. **Dynamic Linking.** By contrast, the dynamic linking process, as depicted in the figure below, creates two separate pieces of binary code, the binary main program and a binary library file.

[25] To the extent it is necessary to provide some background regarding compilation of Java programs, I could do so at trial; however, I assume that the jury will have been provided this background prior to my testimony.



88. As depicted in the following diagram, rather than a single monolithic executable, the complete functionally of the program is provided by a combination of the main executable and the linked library. During the execution of the program, the main program calls upon the dynamically linked library and provides the library with any required data or instructions, if and when the library's functionality is needed. The library then completes execution of the required functionality and returns any data requested by the main program.


Executing a Dynamically Linked Program
Program (.exe)
Compiled Instructions
Compiled function call to linked Library
Compiled Instructions
Compiled function call to linked Library
Compiled function call to linked Library
Library (.dll)

89. The Java Language Specification provides a definition of linking:

> Linking is the process of taking a binary form of a class or interface type and combining it into the runtime state of the Java virtual machine, so that it can be executed. A class or interface type is always loaded before it is linked.

Java Language Specification at 314.

90. The Java Specification also enumerates three different activities involved in linking: verification, preparation, and resolution of symbolic reference. *Verification* refers to the process by which the JVM confirms the structural integrity of a loaded class. *Preparation* refers to the process by which the JVM creates and sets default values for the fields (class variables and constants) of the class or interface being loaded. *Resolution* refers to the process by which references to separate class files included in the loaded class file are confirmed and optimized.

91. The Java Virtual Machine Specification[26] includes the same definition for linking, also describing the JVM's role in loading, linking, and initializing Java class files:

> The Java Virtual Machine dynamically loads, links and initializes classes and interfaces. Loading is the process of finding the binary representation of a class or interface type with a particular name and *creating* a class or interface from that binary representation. Linking is the process of taking a class or interface and combining it into the run-time state of the Java Virtual Machine so that it can be executed.

Java Virtual Machine Specification at Ch. 5.[27]

92. Separately executing applications that are not "linked" as described above can often share data through one or more forms of interprocess communication ("IPC") such as pipes, sockets, shared memory, and system calls. The term "linking" is sometimes used broadly to include forms of IPC and other means by which two pieces of software communicate using an API.

IV. BACKGROUND REGARDING JAVA

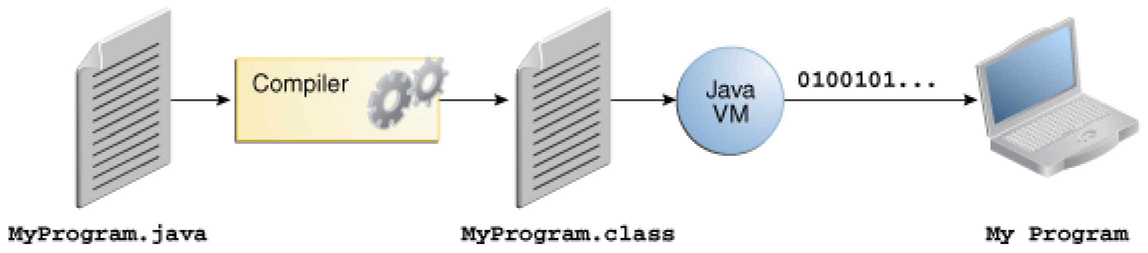
A. Java

93. The term "Java" can refer to many things, including a Java *programming language* or a *Java platform*. The Java Language Specification[28] describes the programming language as a general-purpose, class-based, object-oriented language. The specification distinguishes between "compile-time" and "run time" events and errors. "Run-time activities include loading and linking of the classes needed to execute a program, optional machine code generation and dynamic optimization of the program, and actual program execution." *Id at 1.*

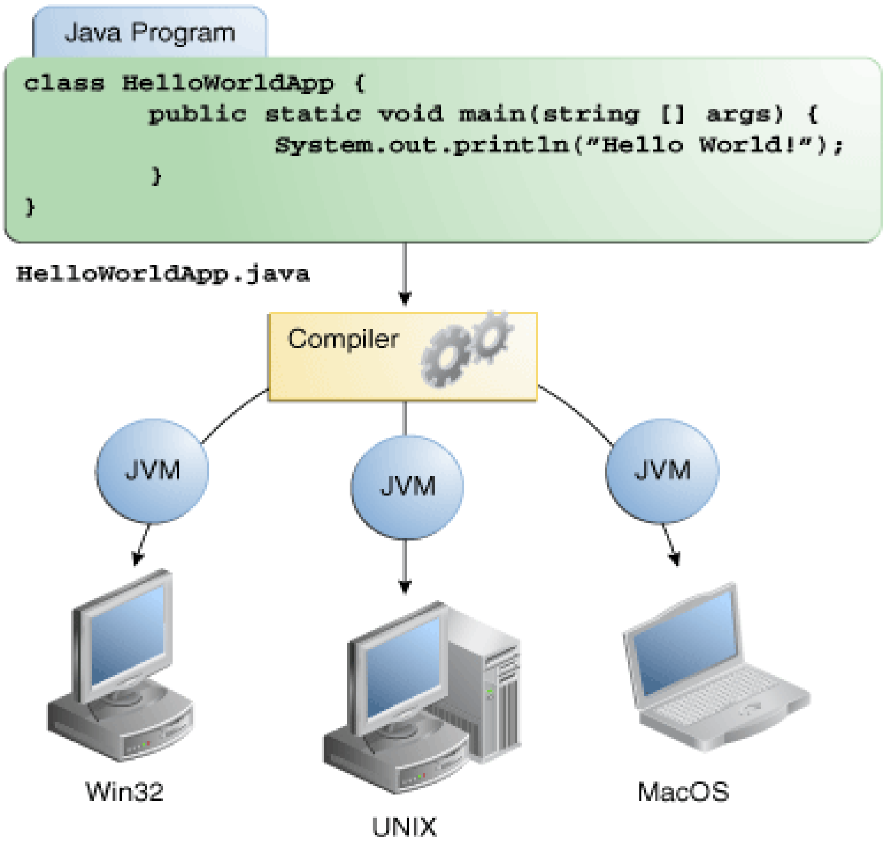
[26] Tim Lindholm, Frank Yellin, The Java Virtual Machine Specification (2nd ed.) available at: https://docs.oracle.com/javase/specs/jvms/se6/html/VMSpecTOC.doc.html.
[27] Ch. 5 HTML version of the Java Virtual Machine Specification (2nd ed.) is available at: https://docs.oracle.com/javase/specs/jvms/se6/html/ConstantPool.doc.html.
[28] James Gosling, Bill Joy, Guy Steele, Gilad Bracha, *The Java Specification* (3rd ed. 2005) available at: http://docs.oracle.com/javase/specs/index.html.

94. Software written in the Java programming language is normally written and saved in plain text files often saved in files with a `.java` or similar extension. Java source code files are then compiled into bytecode, the compiled machine language to be interpreted by a compatible Java Virtual Machine ("JVM"), which "dynamically loads, links, and initializes classes and interfaces."[29] Bytecode is typically stored in the "class file" format. Each class file contains the definition of a single Java class or interface. [30]

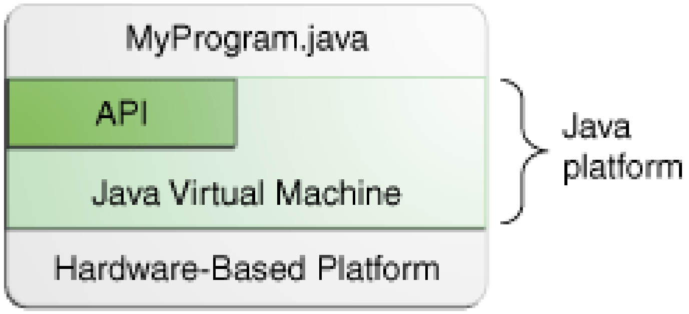
95. A Java platform consists of a JVM and the Java Application Programming Interface ("Java API"), that latter consisting of a set of predefined classes or "Java Core Libraries" providing functional support to executing Java programs and the JVM. Implementations of the Java platform are guaranteed to support the Java programming language, the JVM, and the Java API.[31] The actual code implementing a JVM typically differs from one "host environment" to another.[32] Despite variance in the software implementing a JVM, the same platform-independent bytecode can be used to execute a Java program on any device on which a compatible JVM and Java API have been implemented and installed. These principles are illustrated in the following simple diagrams:



[29] Tim Lindholm, Frank Yellin, *The Java Virtual Machine Specification* , supra n. 26.
[30] Tim Lindholm, Frank Yellin, *The Java Virtual Machine Specification* , supra n. 26..
[31] Sheng Liang, *The Java Native Interface, Programmer's Guide and Specification* (1999) at 4.
[32] The "host environment" includes the host operating system, a set of native libraries, and the CPU instructions set. *See* Sheng Liang, *The Java Native Interface, Programmer's Guide and Specification* (1999) at 4.



```
class HelloWorldApp {
        public static void main(string [] args) {
                System.out.println("Hello World!");
        }
}
```



The Java Tutorials: About the Java Technology (*available at* https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html).

96. Occasionally, Java programs may need to work closely with other software developed and compiled for use in a specific host environment (referred to as "native code"). For example, developers of Java programs that execute on a JVM developed for the Linux operating system may desire to use code developed specifically for use is Linux. The Java Native Interface (or "JNI"), implemented as part of a JVM, is a two-way interface that allows Java programs to

work in cooperation with native code included in *either* native libraries or native applications. The diagram below, taken from the JNI specification, illustrated the role of the JNI in executing of a Java applications.[33]

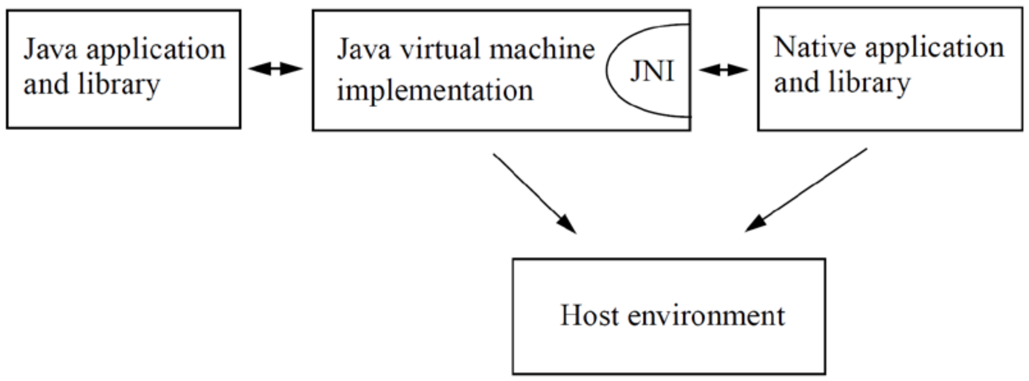

Java application and library
Java virtual machine implementation
JNI
Native application and library
Host environment

Figure 1.1 Role of the JNI

Id at 5.

97. Execution of a Java application begins when the JVM loads and begins interpreting the application's bytecode class file The application class file may refer to (or "call") functionality implemented in a separate class file included in the Java Core Libraries separately made available to the JVM. The JVM is tasked with finding, loading, and linking the class file referenced by the execution Java application. The JVM interprets this referenced class file, providing the functionality requested by the referencing application, before unloading the referenced class file and returning control to the original Java application. The Java Virtual Machine Specification refers to this dynamic loading and unloading of the operational bytecode as "dynamic linking."[34]

[33] Sheng Liang, *The Java Native Interface, Programmer's Guide and Specification* (1999) at 5.
[34] Tim Lindholm, Frank Yellin, *The Java Virtual Machine Specification* (1997) at 18.

B. Relationship with C, C++, and C#.

98. Originally named "Oak," Java was conceived by then Sun employees James Gosling, Patrick Naughton, Christ Warth, Ed Farnk, and Mike Sheridan in 1991. Development of Java was primarily motivated by the need for a platform-independent programming language that could be used to develop software to be embedded on consumer electronic devices, such as home appliances. The same platform-independent features of Java also made it useful for the development of applications written for the internet, networks, and web browsers.

99. As explained in Herbert Schildt's *Java: The Complete Reference*,[35] published by Oracle Press, Java purposefully derives much of its character from the C and C++ languages, which were well known to computer programmers at the time of Java's development and release. As noted by Schildt, "the similarities with C++ are significant, and if you are a C++ programmer, then you will feel right at home with Java." *Id.* at 7. Further, "if you are an experienced C++ programmer, moving to Java will require very little effort. Because Java inherits the C/C++ syntax and many of the object-oriented features of C++, most programmers have little trouble learning Java." *Id* at 11. Schildt discussed some of Java's similarities with C and C++ in greater detail:

> Java is related to C++, which is a direct descendant of C. Much of the character of Java is inherited from these two languages. From C, Java derives its syntax. Many of Java's object oriented features were influenced by C++.

Id at 3.

> Java derives much of its character from C and C++. This is by intent. The Java designers knew that using the familiar syntax of C and echoing the object-oriented features of C++ would make their language appealing to the legions of experienced C/C++ programmers.

[35] Herbert Schildt, *Java – The Complete Reference* 3-11 (9th Ed. 2014).

> In addition to the surface similarities, Java shares some of the other attributes that helped make C and C++ successful. First, Java was designed, tested, and refined by real, working programmers. It is a language grounded in the needs and experiences of the people who devised it. Thus, Java is a programmer's language. Second, Java is cohesive and logically consistent. Third, except for those constraints imposed by the Internet environment, Java gives you, the programmer, full control.

Id at 7.

100. Similarities between Java, C, and C++ have been noted by many others including Jeff Friesen, author of *Learn Java for Android Development*[36] who explains that "Java's syntax (rules for combining symbols into language features) is partly patterned after the C and C++ languages to shorten the learning curve for C/C++ developers" and, more specifically the "[m]any of Java's reserved words are identical to their C/C++ counterparts . . . (catch, class, public, and try are examples)" and Java "supports character, double precision floating-point, floating-point integer, long integer, and short integer primitive types, and via the same char, double, flot, int, long, and short reserved words." Friesen at 2-3.

101. Java has similarly influenced computer programming languages developed after the release of Java. For example, the C# programming language created by Microsoft to support the .NET framework "is closely related to Java. For example, both share the same general syntax, support distributed programming, and utilize the same object model." Schildt at 8.

V. BACKGROUND REGARDING ANDROID

A. The Android Platform

102. Android is a software platform (or "stack") comprised of a series of integrated software components that are themselves comprised of many different open-source software packages. Those packages are generally associated with a particular open-source community or project (such as SQLite) that collaborates to develop, release, and even support the package.

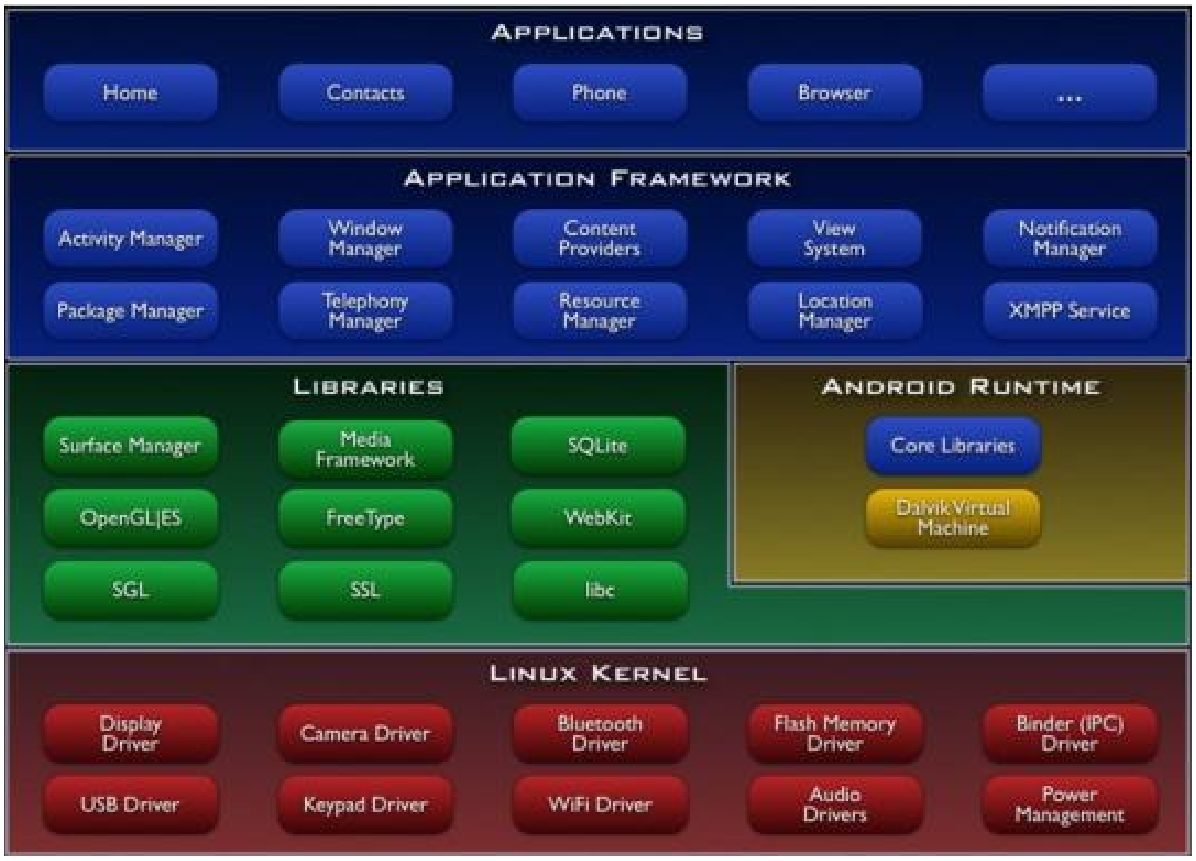
[36] Jeff Friesen, *Learn Java for Android Development* 1-6 (2nd Ed. 2013).

103. The diagram below was published for the Android development community and depicts logically distinct components (or "layers") of the Android platform, and identifies some of the packages included within each component. The components of the depicted stack include: (a) certain basic Android applications with which Android device end users interact; (b) the "Application Framework" comprised of a series of native applications and native libraries that provide functional support to Android applications; (c) the "Android Runtime," comprised of either the Dalvik virtual machine (or "DVM") or the Android Runtime (or "ART")[37] (which execute Dalvik bytecode included in Android applications, Android Core Libraries, and certain Native Libraries and Application Framework libraries written in Java); (d) a series of services and software libraries that provide functional support to Android applications, the Application Framework, and the Android Runtime referred to as "Libraries" in the diagram as " (the "Native Libraries"); and (e) a version of the Linux operating system ("Linux"), which includes both the Linux kernel and a series of hardware and software drivers and other software kernel modules providing functional support to the Linux kernel. The hardware and software drivers bundled with the kernel for a particular Android mobile device depends on the specific hardware components included in that device (*e.g.*, camera, keypad, Bluetooth and Wi-Fi transceivers, and audio processor).

104. Android application source code written in Java can be compiled using the Android SDK to produce an Android package (or "APK"), an archive file including the entire contents of an Android application and used by Android devices to install the Android application. The Java source code is compiled into Dalvik bytecode stored in `.dex` files containing multiple classes to be interpreted by the DVM or ART. Upon installation or during

[37] *See* Art and Dalvik (*available at* https://source.android.com/devices/tech/dalvik/).

execution, the `.dex` files are either interpreted or compiled into native machine code by the DVM or ART. Each Android application is, by default, assigned its own instance (copy) of the runtime (DVM or ART) and a unique Linux user ID used by the Android Platform. Each Android application stack (including the Android application bytecode, DVM/ART, and Android Core Libraries) executes within a separate Linux process.



B. Open-Source Licenses Applicable to Android Components

105. **Linux Kernel.** Linux comprises the Linux kernel and a collection of drivers and other software "kernel modules" that provide functional support to the executing Linux kernel. The Linux kernel is licensed for public use under the GPL-2.0 license with system exceptions.[38]

[38] *See* Linux kernel COPYING file (*available at* https://www.kernel.org/pub/linux/kernel/COPYING) ("NOTE! This copyright does *not* cover user programs that use kernel services by normal system calls - this is merely considered normal use of the kernel, and does *not* fall under the heading of 'derived work'.") (emphasis in original).

The drivers and software packages bundled with the Linux kernel for a particular device are often licensed under the GPL-2.0 license or another GPL-compatible open-source software license. The remainder of the Android platform (depicted above the Linux kernel in the diagram above) executes in the "user space" (*aka* userland), which is part of the computer memory set aside for the execution of Linux applications.

106. **Android Runtime.** The Android Runtime includes either the DVM or ART developed by Google and licensed for public use under the Apache-2.0 license. Past and current distributions of the Android Core Libraries (referred to herein as the "Harmony distributions") include the following packages:

a) 51 Java API packages from the Apache Harmony project incorporated under the Apache-2.0 license which include:

- o 37 Java API packages that I understand to be the subject of this case (the "accused Harmony-based Java API packages");[39] and
- o 14 additional Java API packages that I understand not to be the subject of an infringement claim (the "additional Harmony-based Java API packages");[40] and

[39] I understand that the following 37 API packages are accused:

java.awt.font	java.nio.channels.spi	java.util	javax.net.ssl
java.beans	java.nio.charset	java.util.jar	javax.security.auth
java.io	java.nio.charset.spi	java.util.logging	javax.security.auth.callback
java.lang	java.security	java.util.prefs	javax.security.auth.login
java.lang.annotation	java.security.acl	java.util.regex	javax.security.auth.x500
java.lang.ref	java.security.cert	java.util.zip	javax.security.cert
java.lang.reflect	java.security.interfaces	javax.crypto javax.sql	
java.net	java.security.spec	javax.crypto.interfaces	
java.nio	java.sql	javax.crypto.spec	
java.nio.channels	java.text	javax.net	

b) Android API packages developed and owned by Google and licensed for public use under the Apache-2.0 license (the "Android API packages").[41]

107. As described and analyzed below, certain more recent Android source-code distributions now include Java API packages licensed from the OpenJDK project under the GPL-2.0-CE license instead of Java API packages licensed from the Apache Harmony project under the Apache-2.0 license.

108. **Native Libraries.** The Native Libraries include numerous open-source runtime services and libraries that provide support to executing Android applications. Those libraries authored by Google are licensed by the Android project under the Apache-2.0 license. Most other packages are licensed under one or more permissive licenses such as the Apache-2.0 license, MIT, or BSD licenses. Many of the included Native Libraries (such as SQLite) are well-known to software developers and frequently used to develop both open-source and closed-source software packages. The Native Libraries are dynamically loaded at runtime as needed by Android applications.

109. **Application Framework.** The Application Framework includes a series of runtime services and libraries that are essential to the operating Android platform but which few Android applications directly access. Application Framework packages created and developed by Google are licensed to the Android project under the Apache-2.0 license. Most other packages

[40] I understand that the following additional API packages were originally accused of infringement, but Oracle chose not to assert copyright infringement, in some cases because Oracle uses these packages under license from third parties or allows third parties to utilize these packages under permissive terms:

java.math	javax.xml	javax.xml.transform	javax.xml.validation
java.util.concurrent	javax.xml.datatype	javax.xml.transform.dom	javax.xml.xpath
java.util.concurrent.atomic	javax.xml.namespace	javax.xml.transform.sax	
java.util.concurrent.locks	javax.xml.parsers	javax.xml.transform.stream	

[41] *See* Package Index (*available at* http://developer.android.com/reference/packages.html).

are licensed under one or more permissive licenses (*e.g.*, the Apache-2.0 license, MIT, or BSD licenses).

110. Executing Android applications typically interact with the Application Framework and Native Libraries through a native interface provided by the DVK or ART. The native interface (as with the JNI provided by a JVM) enables Android applications to dynamically link with the native Application Framework and Native Libraries or to communicate with a separately executing service or program using one or more forms of interprocess communication ("IPC").

VI. ANALYSIS REGARDING OPENJDK AND ANDROID

A. OpenJDK

111. Sun released components of the reference implementation of Java SE 6 under the GPL-2.0 and GPL-2.0-CE licenses and established OpenJDK projects focusing on different packages and components within the Java platform.

112. OpenJDK projects accept contributions from many different third-party contributors, including employees of companies such as IBM, RedHat, Apple, Goldman Sachs, Twitter, and SAP—as well as employees of Google itself.

113. OpenJDK consists primarily of the Java compiler, JVM, and the Java Core Libraries. Most of the OpenJDK JVM is published under the GPL-2.0 license. The OpenJDK Java compiler, Java Core Libraries, and those parts of the JVM that provide public APIs are published under the GPL-2.0-CE license.

114. OpenJDK is included within many major Linux distributions such as Ubuntu, Fedora, Red Hat Enterprise Linux (RHEL), Debian, openSUSE, and CentOS. The table below identifies some of the commercial products that include Oracle's OpenJDK software in otherwise closed-source commercial products:

Manufacturer	Product	Notices link
Amazon Web Services	Management Portal for vCenter License Agreement	https://aws.amazon.com/ec2/vcenter-portal/license/
Azul	Zing, Zulu	Azul Zing Licenses and Copyrights
Caseta Wireless	SmartBridge	Lutron Smart Bridge Open Source Information (smart bridge.txt)
Cisco	TelePresence Management Suite Provisioning Extension 1.4; Cisco Modelling Labs; Cisco WebEx Social 3.0(1) SR2; Jabber Guest 10.0	See "Open Source Used in Cisco WebEx Social 3.0(1) SR2"; "Open Source Used in Cisco Jabber Guest 10.0"
HP	Helion Development Platform	See "HP Helion Development Platform Open Source and Third-Party Software License Agreements"
HP	OneView	See "Additional License Authorizations for HP OneView"
IBM	IBM BigInsights 4.0.0	https://www-304.ibm.com/support/knowledgecenter/SSPT3X_4.0.0/com.ibm.swg.im.infosphere.biginsights.install.doc/doc/c0057609.html
IBM	StoredIQ 7.5.0.1	IBM StoreIQ Third-Party Software Guide Version 7.5.0.1
myEarlySense	GS-OTS-SLEEPsense	http://www.myearlysense.com/gs-ots-sleepsense-android-2167777-v1/
Polycom	RealPresence Resource Manager 8.3.1	See Polycom RealPresence Resource Manager "Offer of Source for Open Source Software" version 8.3.1 | March 2014 | 3725-752108.00D1
Pivotal	SQLFire 1.1.2 GA	http://download.pivotal.io/sqlfire/1.1.2/open_source_licenses-Pivotal_SQLFire_1.1.2.txt

B. OpenJDK-Based Android Code

115. As noted above, the publicly available source code for the Android Core Libraries have included the following packages since the first release of Android:

a) 51 Java API packages based on the Apache Harmony project incorporated in Android under the Apache-2.0 license comprised of:

- o the 37 accused Harmony-based Java API packages; and
- o the 14 additional Harmony-based Java API packages not accused of infringement,[42] as well as

b) the Android API packages developed and owned by Google and licensed for public use under the Apache-2.0 license.[43]

116. On Dec. 24, 2015, Google publicly released source code for a modified version of the Android Runtime.[44] That version is derived from 37 Java API packages from the OpenJDK project that are licensed under the GPL-2.0-CE license in place of the 37 API packages previously derived from the Apache Harmony project that were licensed under the Apache-2.0 license. Put another way, the table below summarizes the collections of packages in what are hereinafter referred to as the "Harmony distribution" and "OpenJDK distribution":

Distribution	**Included in Android Core Libraries**
Harmony distribution	• accused Harmony-based Java API packages (Apache-2.0 license) • additional Harmony-based Java API packages (Apache-2.0 license) • Android API packages (Apache-2.0 license)
OpenJDK distribution	• OpenJDK-based Java API packages (GPL-2.0-CE license) • additional Harmony-based Java API packages (Apache-2.0 license) • Android API packages (Apache-2.0 license)

117. In sum, there is relatively little difference between the Harmony and OpenJDK distribution other than swapping the accused Harmony-based Java API packages with the OpenJDK-based Java API packages—specifically, the 37 Java API packages accused of

[42] These API packages are all listed above.
[43] These packages are identified by way of the link provided above.
[44] See https://android.googlesource.com/platform/libcore/+/5d52b7f03d93f3baf03fe37e3d21812e5f9cd361

infringement. While the Android Core Libraries implemented using the OpenJDK-based Java API packages use the same method declarations, the OpenJDK-based API packages include separately developed software licensed under different terms. Android Core Libraries other than the OpenJDK-based Java API packages remain unchanged and Android API packages and additional Harmony-based Java API packages continue to be licensed under the Apache-2.0 license. The most significant difference between the two distributions is the licensing of the included OpenJDK-based Java API packages under the GPL-2.0-CE license, whereas the 37 accused Harmony-based Java API packages they replaced were licensed under the Apache-2.0 license.

C. Linking to OpenJDK-based API Libraries

118. In my opinion, the Android Runtime loading, linking, and executing the OpenJDK-based API packages licensed under the GPL-2.0-CE by or with Android applications, the Native Libraries, other parts of the Android Runtime, other Android Core Libraries, the Application Framework or any other portion of the Android platform will not result in any components other than the OpenJDK-based Java API packages becoming subject to the GPL-2.0 license, the GPL-2.0-CE license, or any other copyleft license. Put another way, the use of the OpenJDK-based API packages will not have a copyleft effect on other software in the Android platform. Instead, the license's copyleft effects are limited to the OpenJDK-based API packages themselves. Like other weak-copyleft licenses, the GPL-2.0-CE imposes a copyleft effect on distributed ***modifications*** to the OpenJDK-based Java API packages, but not to software linking with those packages.

119. As discussed above, an Android application is, as a general matter, loaded in a separate process with its own instance (copy) of the Android Runtime and Android Core Libraries and copies of any Native Libraries or Application Framework libraries used by the

Android application are loaded and linked with the Android application by the Android Runtime. The Android Runtime and Android Core Libraries included on an Android device consist of a series of separate files in the executable and linking (or "ELF") format.[45] Specifically, the OpenJDK-based API packages are included in an ELF file that is separate and distinct from the ELF files that implement the Android Runtime, other Android Core Libraries, Native Libraries, and Application Framework. Moreover, the ELF file for the OpenJDK-based API packages is compiled from a `.dex` bytecode file that is separate and distinct from other `.dex` bytecode files (for example, for the additional Harmony-based Java API packages and Android API packages). The ELF file containing the OpenJDK-based API packages is dynamically loaded during the loading and execution of Android applications by the Android Runtime, meaning that the OpenJDK-based API packages are not combined in the same process with the Android application until the Android program is actually executed (*i.e.*, at runtime). As an Android application executes, the Android Runtime (whether DVM or ART) dynamically links the application with numerous other packages (including at least some of the OpenJDK-based Java API packages) and services. Put another way, the Android Runtime, OpenJDK-based Java API packages, and other Android Core Libraries are needed at runtime by the Android application.

120. As also noted above, distributing software that (a) dynamically or statically links with GPL-2.0-CE-licensed libraries and/or (b) communicates with GPL-2.0-CE-licensed software through one or more forms of IPC will <u>not</u> result in the linked or communicating software becoming subject to the copyleft effects of the GPL-2.0-CE license. Because software combinations with the OpenJDK-based Java API libraries are limited to either linking or one or more forms of IPC, Android's use of the OpenJDK-based Java API packages does not have a

[45] An ELF file results from compiling a `.dex` bytecode into the machine code of the host Android device.

copyleft effect on applications executing on the Android platform, other parts of the Android Runtime, other Android Core Libraries, or any other part of the Android platform. Put another way, the distribution of the OpenJDK-based Java API libraries as included by Google in the Android software stack will not result in obligations to distribute any of the *other* software included in or executing on the Android Platform under the GPL-2.0-CE license.

121. Notably, my opinion regarding the effect of linking Android application, library, and platform code with the OpenJDK-based Java APIs is consistent with opinions held by other influential members of the free software community. For example, Bradley Kuhn, President of the Software Freedom Conservancy and Free Software Foundation Board Member, recently opined on Google's public release of the source code including OpenJDK-based Java API packages as follows:

> First, If you think the ecosystem shall collapse because "pure GPL has moved up the Android stack", and "it will soon virally infect everyone" with copyleft (as you anti-copyleft folks love to say) your fears are just unfounded. Those of us who worked in the early days of reimplementing Java in copyleft communities thought carefully about just this situation. At the time, remember, Sun's Java was completely proprietary, and our goal was to wean developers off Sun's implementation to use a Free Software one. We knew, just as the early GNU developers knew with libc, that a fully copylefted implementation would gain few adopters. So, the earliest copyleft versions of Java were under an extremely weak copyleft called the "GPL plus the Classpath exception". Personally, I was involved as a volunteer in the early days of the Classpath community; I helped name the project and design the Classpath exception. (At the time, I proposed we call it the "Least GPL" since the Classpath exception carves so many holes in strong copyleft that it's less of a copyleft than even the Lesser GPL and probably the Mozilla Public License, too!)
>
> […]
>
> So, how is incorporating Oracle's GPL-plus-Classpath-exception'd JDK different from having an Apache-licensed Java userspace? It's not that much different! Android redistributors already have strong copyleft obligations in kernel space, and, remember that Webkit is LGPL'd;

> there's also already weak copyleft compliance obligations floating around Android, too. So, if a redistributor is already meeting those, it's not much more work to meet the even weaker requirements now added to the incorporated JDK code. I urge you to ask anyone who says that this change will have any serious impact on licensing obligations and analysis for Android redistributors to please prove their claim with an actual example of a piece of code added in that commit under pure GPL that will combine in some way with Android userspace applications. I admit I haven't dug through the commit to prove the negative, but I'd be surprised if some Google engineers didn't do that work before the commit happened.
>
> You may now ask yourself if there is anything of note here at all. There's certainly less here than most are saying about it. In fact, a Java industry analyst (with more than a decade of experience in the area) told me that he believed the decision was primarily technical. Authors of userspace applications on Android (apparently) seek a newer Java language implementation and given that there was a reasonably licensed Free Software one available, Google made a technical switch to the superior codebase, as it gives API users technically what they want while also reducing maintenance burden. This seems very reasonable. While it's less shocking than what the pundits say, technical reasons probably were the primary impetus.
>
> So, for Android redistributors, are there any actual licensing risks to this change? The answer there is undoubtedly yes, but the situation is quite nuanced, and again, the problem is not as bad as the anti-copyleft crowd says. The Classpath exception grants very wide permissions. Nevertheless, some basic copyleft obligations can remain, albeit in a very weak-copyleft manner. It is possible to violate that weak copyleft, particularly if you don't understand the licensing of all third-party materials combined with the JDK. Still, since you must comply with Linux's license to redistribute Android, complying with the Classpath exception'd stuff will require only a simple afterthought.

See *Sun, Oracle, Android, Google and JDK Copyleft FUD* (http://ebb.org/bkuhn/blog/2016/01/05/jdk-in-android.html) (emphases in original).

122. While my analysis above applies to the recently released OpenJDK-based Java API packages, it is also my opinion that Google could have adopted the same approach when OpenJDK was first open-source licensed pursuant to the GPL-2.0-CE license (in 2007). For purposes of my analysis, the license terms were not materially different then, and the Classpath

Exception allowed the same degree of linking without "tainting" the linking code. In short, as of 2007, Google could have used the 37 Java API packages—including both the implementing code and method declarations—from the OpenJDK project and licensed under the GPL-2.0-CE license instead of the 37 API packages previously derived from the Apache Harmony project and licensed under the Apache-2.0 license. And, relatedly, it could have otherwise licensed the additional 14 Harmony-based Java API packages and its own Google-developed Android API packages under the Apache-2.0 license, just as it did then and still does today.

VII. CONCLUSION

123. I reserve the right to update and refine my opinions and analyses in light of any additional materials or information that may come to my attention in the future, and to supplement my opinions and analyses as set forth in this report in light of any expert reports submitted by Oracle and in light of any deposition or trial testimony of Oracle's experts.

Executed on the 8th of January, 2016 in Novato, CA.

[illegible]

Andrew Hall

APPENDIX A – CURICULUM VITAE OF ANDREW J. HALL

100 PINE STREET STE 1250, SAN FRANCISCO, CA 94111, (415) 988-9957, AHALL@THEHALLLAW.COM

EXPERIENCE

2015 TO PRESENT **Hall Law** *San Francisco, CA*
PARTNER

Focuses on developing and implementing software commercialization strategies including both commercial and free and open-source software (FOSS) components

Leverages technical background to help clients protect their valuable intellectual property and technical assets, avoid contractual disputes and claims of infringement, and develop effective and efficient policies, practices, and procedures for commercializing their software and other intellectual property assets

2010 TO 2015 **Fenwick & West LLP** *Mountain View, CA & San Francisco, CA*
INTELLECTUAL PROPERTY ATTORNEY

Counseled technology clients on risk management in relationship to the development and commercialization of computer hardware and software

Designed and implemented policies, procedures, and training programs to efficiently minimize the risk to clients and their assets including programs for managing and documenting the use of thousands of free and open-source software packages

2006 TO 2010 **Knobbe Martens Olson & Bear** *Irvine, CA*
INTELLECTUAL PROPERTY ATTORNEY

Engaged in broad intellectual property practice focusing on patent litigation, but also including due diligence, licensing, client counseling, patent and trademark prosecution, and patent valuation

Responsible for daily management of litigation matters for international consumer electronics and medical device companies

EDUCATION

- 2006 University of Chicago Law School *Chicago, IL*
- JURIS DOCTOR

The Ohio State University *Columbus, OH*

-
- 2002

BACHELOR OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

Honors: National Merit Scholar; Tau Beta Pi; Alpha Lambda Delta; Phi Eta Sigma; Golden Key; University College Suma Award

Activities: Editorialist for OSU School of Journalism Newspaper, The Lantern (97-01); Editor, The Lantern (98-00); Adopt-a-School volunteer, tutoring of inner-city children (98-00);

SPEAKING ENGAGEMENTS

Palamida User Group Meeting September 2015 presented jointly with the CEO: *The Latest News & Insights from the OSS World*

Linux Collaboration Summit February 2015: *Open Source Business Models: Making Money By Giving It Away*

Santa Clara Law Open Source Symposium January 2015 Keynote Speaker: *Open Source Licensing and Business Models: Making Money by Giving it Away*

All Things Open October 2014: *Open Source Licensing & Business Models*

Great Wide Open April 2014: *Profiting with Open Source*

Linux Foundation Collaboration Summit March 2014: *An Introduction to FOSS Licensing*

APPENDIX B – DOCUMENTS EXAMINED DURING REPORT PREPARATION

I examined the following documents, published materials, and websites in the course of preparing this report include:

- Jeff Friesen, *Learn Java for Android Development* (2nd Ed. 2013)
- Herbert Schildt, *Java – The Complete Reference* (9th Ed. 2014)
- Budi Kurniawan, *Java 7 – A Beginner's Tutorial* (3rd Ed. 2011)
- Ralph Morelli & Ralph Walde *Java, Java, Java – Object-Oriented Problem Solving* (3rd Ed. 2006)
- James Gosling, Bill Joy, Guy Steele, Gilad Bracha, *The Java Specification* (3rd Ed. 2005).
- Tim Lindholm, Frank Yellin, The Java Virtual Machine Specification (2nd Ed.)
- Sheng Liang, *The Java Native Interface, Programmer's Guide and Specification* (1999)

APPENDIX C – OPENJDK LICENSES

GPL-2.0

The GNU General Public License (GPL)

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS licenseD FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT

LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

One line to give the program's name and a brief idea of what it does.

Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option)any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

"CLASSPATH" EXCEPTION TO THE GPL

Certain source files distributed by Oracle America and/or its affiliates are subject to the following clarification and special exception to the GPL, but only where Oracle has expressly included in the particular source file's header the words "Oracle designates this particular file as subject to the "Classpath" exception as provided by Oracle in the license file that accompanied this code."

Linking this library statically or dynamically with other modules is making a combined work based on this library. Thus, the terms and conditions of the GNU General Public License cover the whole combination.

As a special exception, the copyright holders of this library give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you may extend this exception to your version of the library, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

OPENJDK ASSEMBLY EXCEPTION

The OpenJDK source code made available by Oracle at openjdk.java.net and openjdk.dev.java.net ("OpenJDK Code") is distributed under the terms of the GNU General Public License <http://www.gnu.org/copyleft/gpl.html> version 2 only ("GPL2"), with the following clarification and special exception.

Linking this OpenJDK Code statically or dynamically with other code is making a combined work based on this library. Thus, the terms and conditions of GPL2 cover the whole combination.

As a special exception, Oracle gives you permission to link this OpenJDK Code with certain code licensed by Oracle as indicated at http://openjdk.java.net/legal/exception-modules-2007-05-08.html ("Designated Exception Modules") to produce an executable, regardless of the license terms of the Designated Exception Modules, and to copy and distribute the resulting executable under GPL2, provided that the Designated Exception Modules continue to be governed by the licenses under which they were offered by Oracle.

As such, it allows licensees and sublicensees of Oracle's GPL2 OpenJDK Code to build an executable that includes those portions of necessary code that Oracle could not provide under GPL2 (or that Oracle has provided under GPL2 with the Classpath exception). If you modify or add to the OpenJDK code, that new GPL2 code may still be combined with Designated Exception Modules if the new code is made subject to this exception by its copyright holder.